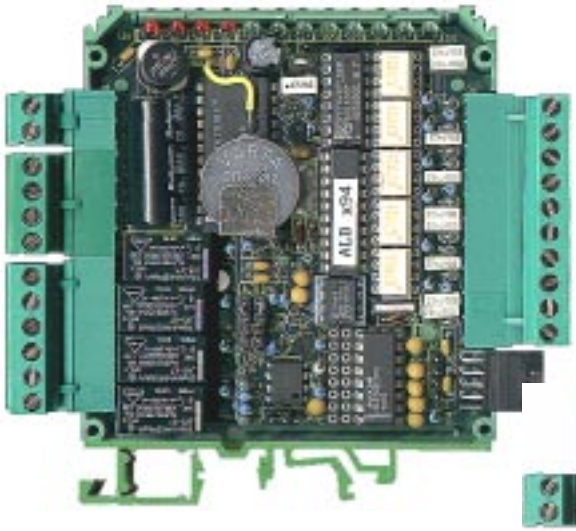
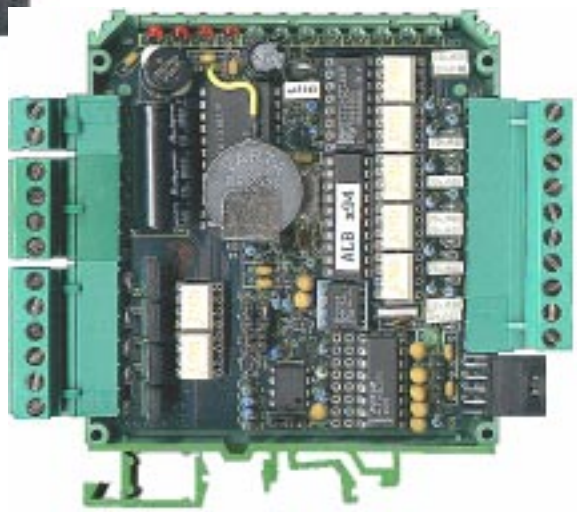
GPC® R/T94

General Purpose Controller
Relays or Transistors; 9 Inputs, 4 Outputs

TECHNICAL MANUAL





grifo®
ITALIAN TECHNOLOGY

Via dell' Artigiano, 8/6
40016 San Giorgio di Piano
(Bologna) ITALY
E-mail: grifo@grifo.it
http://www.grifo.it http://www.grifo.com
Tel. +39 051 892.052 (a.r.) FAX: +39 051 893.661

GPC® R/T 94 Edition 5.40 *Rel. 27 July 2001*

GPC® R/T94

General Purpose Controller
Relays or Transistors; 9 Inputs, 4 Outputs

TECHNICAL MANUAL

Intelligent module **ABACO®** Block Serie M; Size 22.5x82x90 mm; Container for Ω rails type **DIN 46277-1** and **DIN 46277-3**; CPU **89C2051** or **89C4051** with **14MHz** quartz; **2K** or **4K** Bytes **FLASH** and **128** Bytes **SRAM**; Code compatible with family 51 µP; **Real Time Clock** with 256 Bytes internal SRAM; **Back Up** circuitry for **RTC and SRAM**, through **Lithium battery**;**Serial E²** up to **1K** Bytes; **9** galvanically isolated **NPN input** lines; Galvanically isolated **external INT signal**; **4 output** lines: **5A relays** in version R, Optocoupled **4A**, 45 Vdc **NPN Darlington** without heat sink and with back EMF protection diode, in version T; **LEDs** to visualize the status of all the I/O signals; I/O signals connection through quick release connectors; 2 TTL I/O signals; **16 bits Timer Counter** register; **A/D Converter** with 11 bits of resolution, conversion time 60 ms, analog input signal in the range 0÷10Vdc, 0÷20 mA or 4÷20 mA; **Serial line** in **TTL**, **RS 232**, **RS 422**, **RS 485** or Current Loop; Possibility of **Idle Mode** or **Power Down Mode**; **High efficiency** switching power supply on board; Supply of galvanically isolated section: +24 Vdc; On board logic **power supply**: **5 Vdc** or 10÷40 Vdc or 8÷24Vac; **Protection** against transients **by TransZorb™**; Telecontrol Firmware featuring ALB (**ABACO®** Link BUS) communication protocol; Wide range of development software available like C Compiler, Assembler, BASIC Compiler BASCOM 8051, HTC-51, etc.

grifo®
ITALIAN TECHNOLOGY

Via dell' Artigiano, 8/6
40016 San Giorgio di Piano
(Bologna) ITALY
E-mail: grifo@grifo.it
http://www.grifo.it http://www.grifo.com
Tel. +39 051 892.052 (a.r.) FAX: +39 051 893.661

GPC® R/T 94 Edition 5.40 *Rel. 27 July 2001*

DOCUMENTATION COPYRIGHT BY grifo® , ALL RIGHTS RESERVED

IMPORTANT

Although all the information contained herein have been carefully verified, **grifo**® assumes no responsability for errors that might appear in this document, or for damage to things or persons resulting from technical errors, omission and improper use of this manual and of the related software and hardware.
grifo® reserves the right to change the contents and form of this document, as well as the features and specification of its products at any time, without prior notice, to obtain always the best product.
For specific informations on the components mounted on the card, please refer to the Data Book of the builder or second sources.

SYMBOLS DESCRIPTION

In the manual could appear the following symbols:

Attention: Generic danger

Attention: High voltage

Trade Marks

abaco®, **GPC**®, **grifo**® : are trade marks of **grifo**®.
Other Product and Company names listed, are trade marks of their respective companies.

GENERAL INDEX

FIGURES INDEX

INTRODUCTION

The use of these devices has turned - IN EXCLUSIVE WAY - to specialized personnel.

The purpose of this handbook is to give the necessary information to the cognizant and sure use of the products. They are the result of a continual and systematic elaboration of data and technical tests saved and validated from the manufacturer, related to the inside modes of certainty and quality of the information.

The reported data are destined- IN EXCLUSIVE WAY- to specialized users, that can interact with the devices in safety conditions for the persons, for the machine and for the enviroment, impersonating an elementary diagnostic of breakdowns and of malfunction conditions by performing simple functional verify operations , in the height respect of the actual safety and health norms.

The informations for the installation, the assemblage, the dismantlement, the handling, the adjustment, the reparation and the contingent accessories, devices etc. installation are destined - and then executable - always and in exclusive way from specialized warned and educated personnel, or directly from the TECHNICAL AUTHORIZED ASSISTANCE, in the height respect of the manufacturer recommendations and the actual safety and health norms.

The devices can't be used outside a box. The user must always insert the cards in a container that rispect the actual safety normative. The protection of this container is not threshold to the only atmospheric agents, but specially to mechanic, electric, magnetic, etc. ones.

To be on good terms with the products, is necessary guarantee legibility and conservation of the manual, also for future references. In case of deterioration or more easily for technical updates, consult the AUTHORIZED TECHNICAL ASSISTANCE directly.

To prevent problems during card utilization, it is a good practice to read carefully all the informations of this manual. After this reading, the user can use the general index and the alphabetical index, respectly at the begining and at the end of the manual, to find information in a faster and more easy way.

CARD VERSION

The present handbook is reported to the following versions:

- Board **GPC® R94**: version **221199** and later.
- Board **GPC® T94**: version **221199** and later.
- Firmware **ALB x94**: version **1.3** and later.

The validity of the bring informations is subordinate to the number of the card release. The user must always verify the correct correspondence among the two denotations. On the card the release number is present in more points both board printed diagram (serigraph) and printed circuit (for example near battery BT1 on solder side). The firmware version is labelled on the microprocessor, or it can be requested by the proper serial command.

GENERAL INFORMATIONS

The GPC® R94 or **T94** belongs to the CPUs **M Serie** 22,5x82x90 mm size, and it is a powerful control **Low-Cost** module capable of operationg in stand-alone mode or as an intelligent peripheral, and/or remoted, in a wider telecontrol or aquisition network. The **GPC® R94** or **T94** module is secured in a plastic mount for connection to **Omega** rails **DIN 46277-1** and **DIN 46277-3**, thereby dispensing with the need of rack and allowing a less costly mounting direct to the electrical control panel. The facility in the performance can be carried on thanks to a wide range of software developement tools with evoluted language which, in an efficient and friendly environment, allow to work at the best using a standard PC. For example with **BASCOM 8051** and **SIM2051** is possible generate the apllications in really fast time and with minimal investments.
The **GPC® R94** or **T94** is available with telecontrol firmware already programmed on the board. In this case throught **ALB** (**Abaco® Link BUS**) it can be used as intelligent module in stand-alone mode or in network mode with a normal serial line. It is so possible, to realize applications with few I/O drived by a remote PC.

- Intelligent **ABACO® BLOCK** module, **M serie**.
- 22.5 x 82 x 90 mm size.
- Plastic mount for connection to **DIN 46277-1** and **DIN 46277-3** Ω rails.
- **14 MHz 89C2051** or **89C4051** CPU.
- **2K bytes** or **2K bytes FLASH** and **128 bytes SRAM**.
- Code compatible with 51 family µP.
- **Real Time Clock** with **256 bytes** of internal **SRAM.**
- **Back Up** circuitery for **SRAM** and **RTC** through **LITHIUM** battery.
- 1K bytes serial **EEPROM**.
- **9 Optocoupled Input TTL** lines.
- **Optocoupled** external **INT** line.
- **4** outputs: **5**A **Relay** in the **R** version.
  Optocoupled **4A**, 45 Vdc **NPN Darlington** without heat sink and with back EMF protection diode, in version T.
- **LEDs** to visualize I/O lines status.
- All I/O connections are available on quick release connectors.
- 16 bits **Timer Counter** registers.
- **1** line 11 bits **A/D Converter** with analog input range = 0÷10Vdc and conversion time = 60 ms.
- 1 Serial line in **TTL, RS 232**, **RS 422**, **RS 485 or Current Loop**.
- Facility of operation in **Idle** or **Power down** mode.
- High efficiency on board switching power supply.
- +24 Vdc for optocoupled power supply.
- **5** or **5÷40 Vdc** or **10÷24 Vac** for on board logic.
- Protection against voltage peaks by **TransZorb**™.
- Wide range of base software and **Developement Tools** enables the use of the card connected to any PC, making a special developement system unnecessary. Among the available programs there are: **"C" Compiler, Assembler, BASIC Compiler BASCOM LT, HTC-51,** etc.

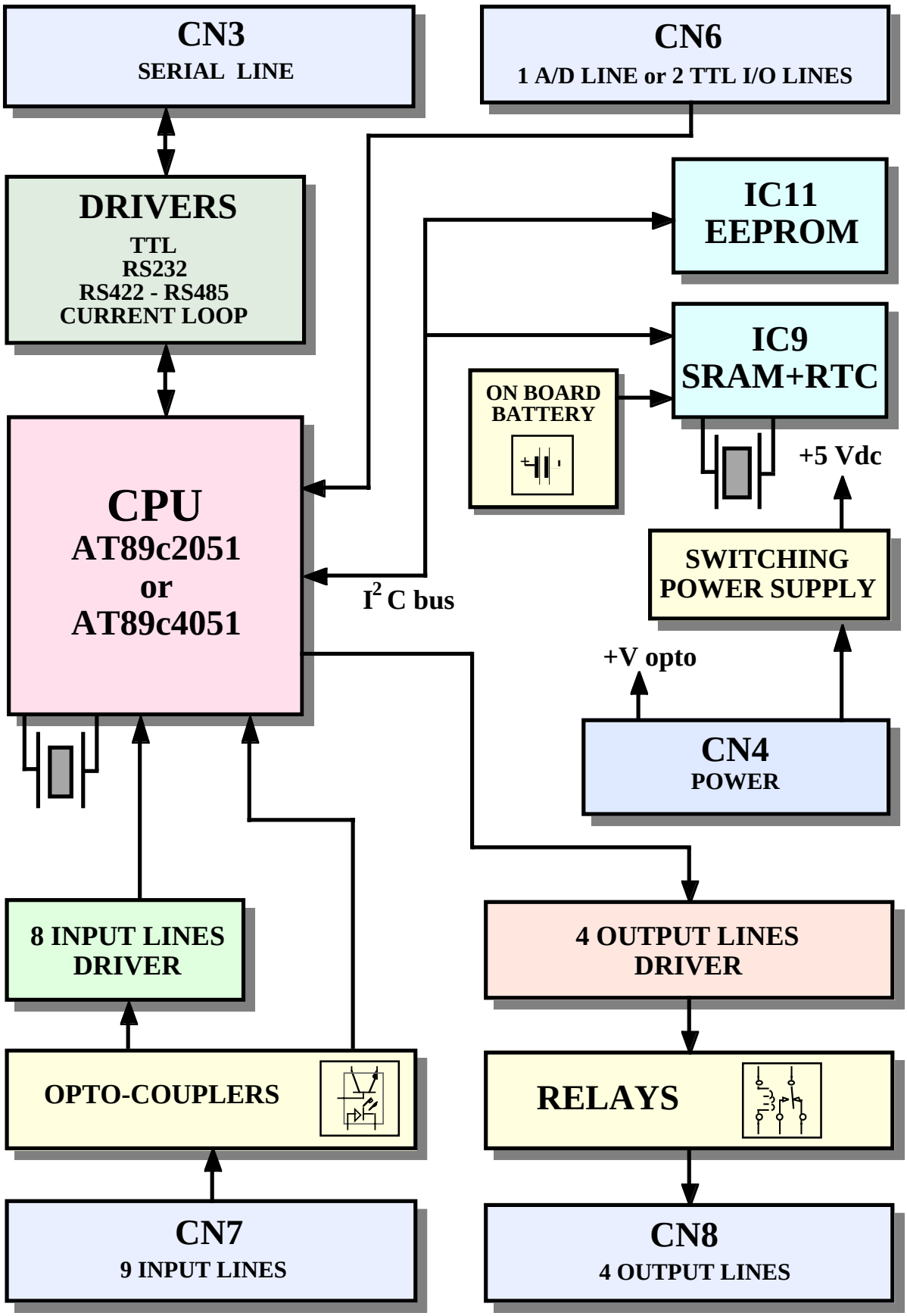


FIGURE 1: GPC® R 94 BLOCK DIAGRAM

Here follows a description of the board's sections and the operations they perform. To easily locate such section on verify their connections please refer to figures 1 and 2.

CPU

The **GPC® R/T94** can use all version of ATMEL microprocessors namely 89c2051 or 89c4051. These 8 bit microprocessors are provided with an internal FLASH EPROM of respectively 2K Bytes or 4K Bytes and are code compatible with the 8051 INTEL and so they have an extended instruction set, fast execution and data manipulation time, easy use of all kind of memory, several hardware peripherals inside and an efficent interrupt management. The most important features of the described microprocessors, are:

µP AT89c2051 AT89c4051:
- 8 bit CPU;
- 2K or 4K bytes of ROM FLASH;
- 128 bytes of internal SRAM;
- 15 programmable I/O lines;
- 2 x 16 bits Timer/Counters;
- 5 interrupts sources;
- 2 priority level for interrupts;
- 1 full duplex serial line;
- 1 analog comparator;
- Low Power Idle and Power Down Modes;

The **GPC® R/T94** by default is delivered with an AT89c4051 already programmed with telecontrol program called **ALB x94**. In this case, thanks to ALB (**ABACO®** Link BUS) comunication protocol, it can be used like an intelligent peripheric, alone or in net by a standard serial line; so even applications with limited number of I/O, driven by a normal PC, could be realized in an easy way. For further information, please refer to specific documentation of the manufacturing company or to appendix B of this manual.

POWER SUPPLY SECTION

One of the most important features of **GPC® R/T94** is its on board power supply circuitery; the card can be powered in two different ways: +5 Vdc (without power supply section) or 10÷24Vac (switching section). The power supply circuit generates the necessary voltages for the card, starting from all the standard industrial source like mains, power transformer, battery, solar cell, etc. The User must provide also a 24 Vdc voltage for the optocoupled section. The power supply type must be specified at the moment of the order. The power supply circuit was designed for reducing the consumption (the microprocessor power down and idle mode is available) and for increasing the electrical noise immunity. Remember that on board there is a protection circuit aganist voltage peaks by **TransZorb™**.

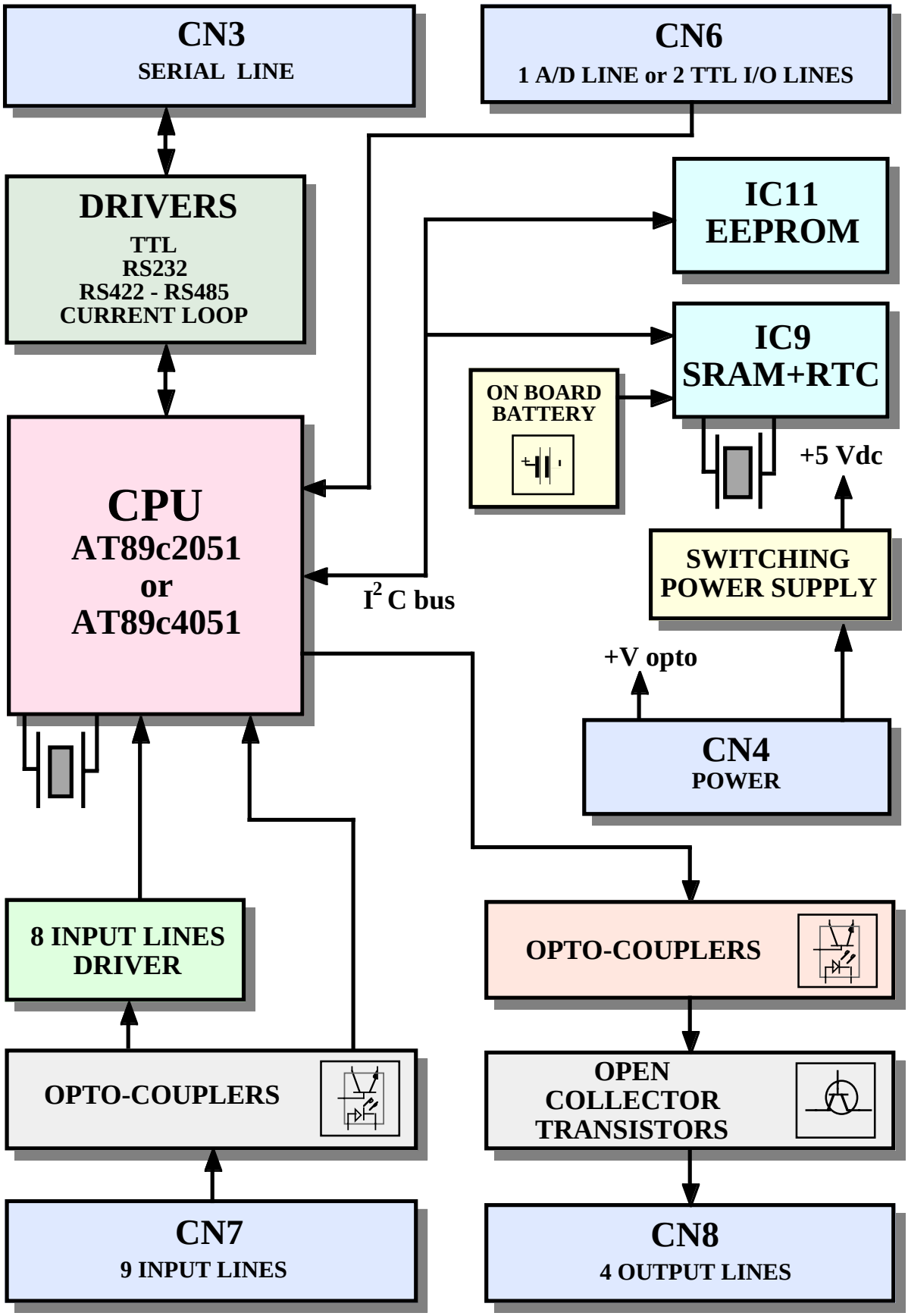


FIGURE 2: GPC® T 94 BLOCK DIAGRAM

CLOCK

Installed on **GPC® R/T 94** boards there are two separated circuitries, based on two different quartzes, to generate clock signals for the microprocessor (14.7456 MHz) and for the Real Time Clock (32.768 KHz). The choice of using two circuits and two separated crystals, has the advantage to change the microprocessor working speed (when best performances are required) without additional changes in software, firmware, etc.

MEMORY DEVICES

The total memory amount that the card can support can be up to 1280 Bytes. Normally the card is equipped with 512 bytes of serial EEPROM and 256 bytes of serial SRAM+RTC. With the on board back up circuit there is the possibility to keep the 256 SRAM+RTC data, also when power supply is failed; in this way the card is always able to maintain parameters, logged data, system status and configuration, etc. without using expensive external UPS. The back up circuit is based on an internal LITHIUM battery.

- **Real Time Clock:** The IC 9 SRAM module, is provided with on board lithium battery and with Real Time Clock which manages time (hours, minutes, seconds) and date (day, month, year, day of the week).

- **Serial EEPROM:** With the IC11 EEPROM module (range 512÷1024 bytes), there is the possibility to keep data also when power supply failed; in this way the card is always able to maintain parameters, logged data, system status and configuration, etc. without using expensive external UPS. This component has a default size of 512 bytes.

For further information about peripheral device please refer to the technical documentation of the manufacturing company.

GPC® R94 OUTPUT SECTION

This section has 4 output lines, available on a comfortable quick release connector, driven by PNP transistor. These components are managed through 4 CPU pins,following the informations explained in the "HARDWARE DESCRIPTION" and "SOFTWARE DESCRIPTION" chapters. Each line has an own LED to show the line status and it drives a 5A N.O. Relay.

GPC® T94 OUTPUT SECTION

This section has 4 output lines, available on a comfortable quick release connector, driven by optocoupler. These components are managed through 4 CPU pins,following the informations explained in the "HARDWARE DESCRIPTION" and "SOFTWARE DESCRIPTION" chapters. Each line is galvanically isolated and has an own LED to show the line status and it can drive a 4A, 45Vdc Darlington transistor that is conected in Open Collector with recovery rectifier.

INPUT SECTION

Throught a 74HCT166 (driven by 3 CPU pins) and one CPU pin the User can read the 9 input lines of the card. Each NPN input lines is optcoupled and its state is shown by a proper LED. The optocoupled section is powered by the +Vopto voltage that is provided by the power supply section. Please remake that if necessary it is possible to acquire the signals of three lines very quickly bypassing the serializer and connecting opportunaly the desired lines. For further informations please contact directly the **grifo®** technical personnel.

FIRMWARE ALB X94

GPC® R/T 94 boards are delivered by default with the firmware ALB x94 (**ABACO® L**ink **BUS**), that allows to manage all the on-board resources through several commands sent to the serial line. A remarkable feature is the implementation of two separated **1-Wire®** protocol communication buses made using the I/O TTL lines. It is so possible to manage two different 1-Wire® devices through the serial line of **GPC® R/T 94** (temperature sensors, memories, Dallas iButton™, etc.).
It is also supported a Master-Slave communication mode that allows to remote control single modules from great distance, realizing a telecontrol network managed by an unique master unit (PC, PLC, **GPC®** serie board, etc.).

A/D CONVERTER AND TTL I/O LINES

GPC® R/T 94 boards are provided with an analog input for A/D conversion, based on a circuitry that, by means of the microprocessor inside analog comparator, can acquire a signal in the range 0÷10 V, 0÷20 mA or 4÷20 mA, with 11 bits of resolution and a maximum conversion time of 60 msec.
This analog input is reachable on a comfortable quick release connector, to interface easily with the external world.
In addition, it is possible to bypass the analog circuitry by moving some jumpers, and connect the microprocessor's pins directly to the connector, obtaining two I/O TTL lines for general purpose or just a direct access to the inputs of the comparator.

SERIAL COMMUNICATION SELECTION

An asyncronous serial line is available on **GPC® R/T94** and it can be buffered in TTL, RS 232, RS 422, RS 485 or Current Loop. By hardware can be selected which one of these electric standards is used, by software the serial line can be programmed to operate with all the standard physical protocols, in fact the bits per character, parity, stop bits and baud rates can be decided by setting opportunes CPU internal registers. For further informations please refer to the manifacturer documentation or to appendix B of this manual.

TECHNICAL FEATURES OF GPC® R94

GENERAL FEATURES OF GPC® R94

On board resources:
9 NPN optocoupled digital inputs
4 N.A. 5 A relays digital outputs
2 TTL input/output
1 NPN external optocoupled interrupt
1 analog comparator
1 A/D Converter signal
1 Real Time Clock
1 serial line TTL, RS 232, RS 422-485 or Current Loop

Memory devices:
IC 9: serial RTC+SRAM 256 bytes
IC 11: serial EEPROM from 512 bytes to 1024 bytes

CPU: Atmel AT89C2051 or AT89C4051

Clock frequence:
14.7456 MHz (CPU)
32.768 KHz (Real Time Clock)

A/D resolution: 11 bits

A/D conversion time: Max 60 ms

PHYSICAL FEATURES OF GPC® R94

Size (W x H x D):
111 x 90 x 22,5 mm
111 x 102 x 22,5 mm (included container for Ω rails)

Mounting: On Ω rails type DIN 46277-1 and DIN 46277-3

Weight: 182 g (basic version)

Connectors:
CN3: 4+4 pins AMP Mod II male 90°
CN4: 4 pins quick release 90°
CN6: 2 pins quick release 90°
CN7: 10 pins quick release 90°
CN8: 6 pins quick release 90°

Female connector for CN3: **grifo®** code: **CKS.AMP8** (kit made of a female AMP Mod II 4+4 8 ways connector and pins to crimp)

Temperature range: from 0 to 50 Centigrad degreeses

Relative humidity: 20% up to 90% (without condense)

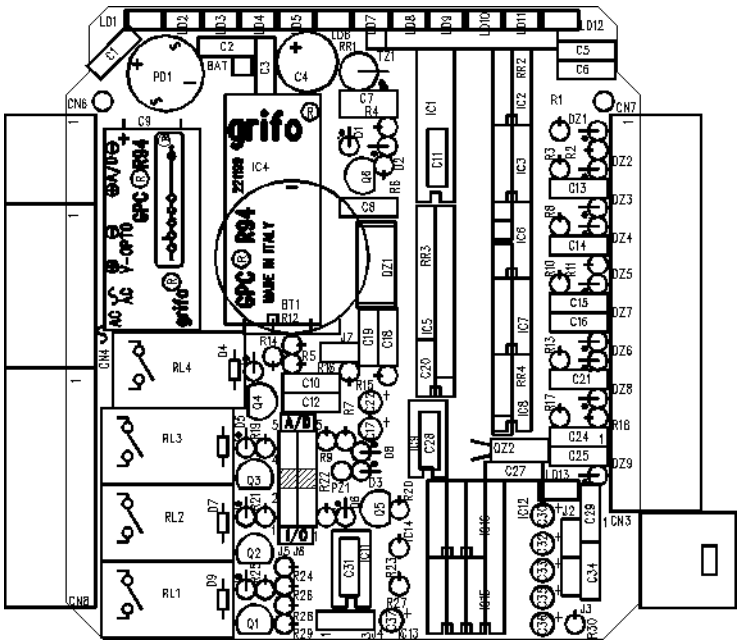


FIGURE 3: COMPONENTS MAP GPC® R94 (COMPONENT SIDE)

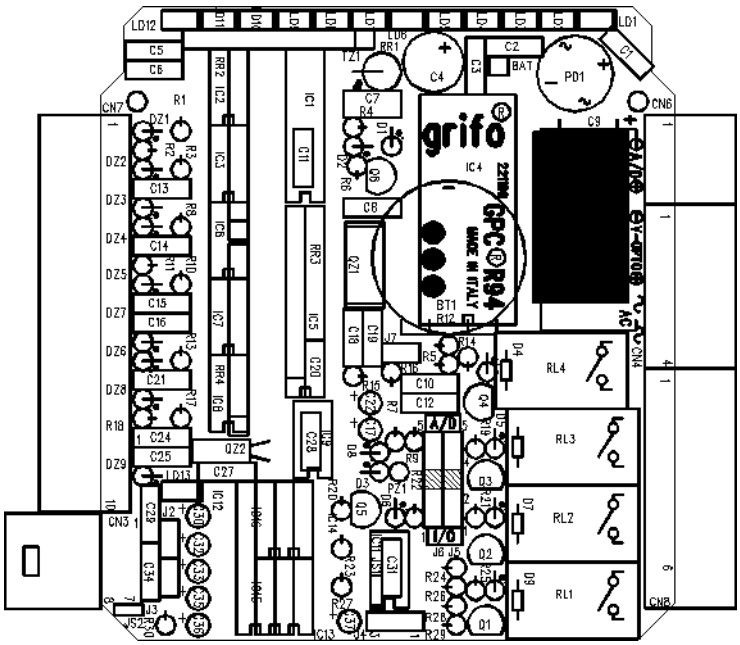


FIGURE 4: COMPONENTS MAP GPC® R94 (SOLDER SIDE)

ELECTRIC FEATURES OF GPC® R94

Version without power supply

Supply voltage:	+5 Vdc	(control logic)
	+24 Vdc	(+V opto)

Version with switching power supply

Input supply voltage:	10÷40 Vdc o 8÷24 Vac	(control logic)
	24 Vdc	(+V opto)
Output supply voltage:	+5 Vdc; 740 mA	
	+Vopto 100 mA	
Current consumption:	260 mA max	(+5 Vdc)
	40 mA max	(+V opto)
Maximum current on relays:	5A	
Maximum voltage on relays:	30 Vdc / 250 Vac	
NPN inputs minimum current:	2 mA	
Back Up current:	3.5 µA	
Back Up on board battery:	3 V; 180 mAh; mod. CR 2032	
Analog input:	0÷10 V; 0÷20 mA; 4÷20 mA	
Analog input impedance:	1 MΩ	
RS422-485 termination network:	Line termination resistor	= 120 Ω
	Positive pull up resistor	= 3,3 KΩ
	Negative pull down resistor	= 3,3 KΩ

TECHNICAL FEATURES OF GPC® T94

GENERAL FEATURES OF GPC® T94

On board resources:
9 NPN optocoupled digital inputs
4 open collector darlington NPN digital outputs, with back EMF protection diode
2 TTL input/output
1 NPN external optocoupled interrupt
1 analog comparator
1 A/D Converter signal
1 Real Time Clock
1 serial line TTL, RS 232, RS 422-485 or Current Loop

Memory devices:
IC 9: serial RTC+SRAM 256 bytes
IC 11: serial EEPROM from 512 bytes to 1024 bytes

CPU: Atmel AT89C2051 or AT89C4051

Clock frequence:
14.7456 MHz (CPU)
32.768 KHz (Real Time Clock)

A/D resolution: 11 bits

A/D conversion time: Max 60 ms

PHYSICAL FEATURES OF GPC® T94

Size (W x H x D):
111 x 90 x 22,5 mm
111 x 102 x 22,5 mm (included container for Ω rails)

Mounting: On Ω rails type DIN 46277-1 and DIN 46277-3

Weight: 171 g (basic version)

Connectors:
CN3: 4+4 pins AMP Mod II male 90°
CN4: 4 pins quick release 90°
CN6: 2 pins quick release 90°
CN7: 10 pins quick release 90°
CN8: 6 pins quick release 90°

Female connector for CN3: **grifo®** code: **CKS.AMP8** (kit made of a female AMP Mod II 4+4 8 ways connector and pins to crimp)

Temperature range: from 0 to 50 Centigrad degreeses

Relative humidity: 20% up to 90% (without condense)

ELECTRIC FEATURES OF GPC® T94

Version without power supply

Supply voltage:	+5 Vdc	(control logic)
	+24 Vdc	(+V opto)

Version with switching power supply

Input supply voltage:	10÷40 Vdc o 8÷24 Vac	(control logic)
	24 Vdc	(+V opto)
Output supply voltage:	+5 Vdc; 930 mA	
	+Vopto 100 mA	
Current consumption:	70 mA max	(+5 Vdc)
	40 mA max	(+V opto)
Maximum current on Transistor:	4A not continuative	(*)
Maximum voltage on Transistor:	45 Vdc	(*)
Maximum power on Transistor:	1.25 W	(*)
NPN inputs minimum current:	2 mA	
Back Up current:	3,5 µA	
Back Up on board battery:	3 V; 180 mAh; mod. CR 2032	
Analog input:	0÷10 V; 0÷20 mA; 4÷20 mA	
Analog input impedance:	1 MΩ	

RS422-485 termination network:	Line termination resistor	=	120 Ω
	Positive pull up resistor	=	3,3 KΩ
	Negative pull down resistor	=	3,3 KΩ

(*) These values are refferred to a 20 °C work temperature

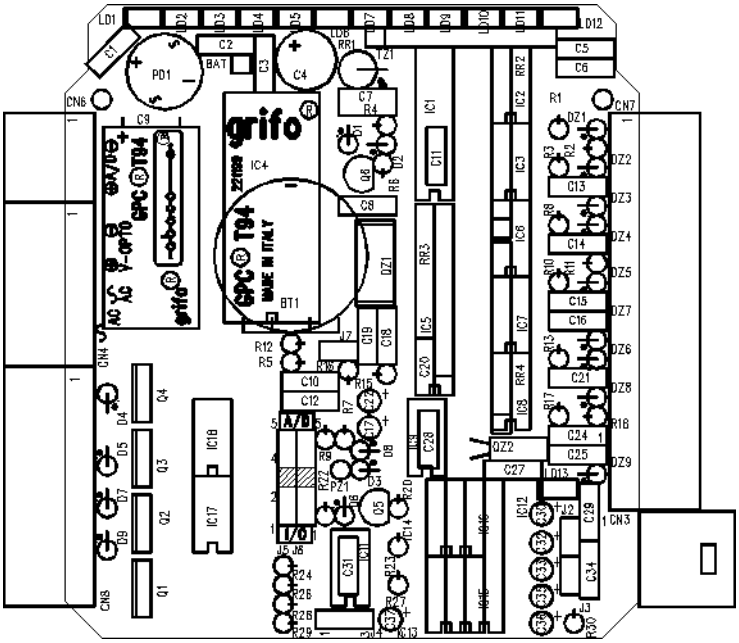


FIGURE 5: COMPONENTS MAP GPC® T94 (COMPONENT SIDE)

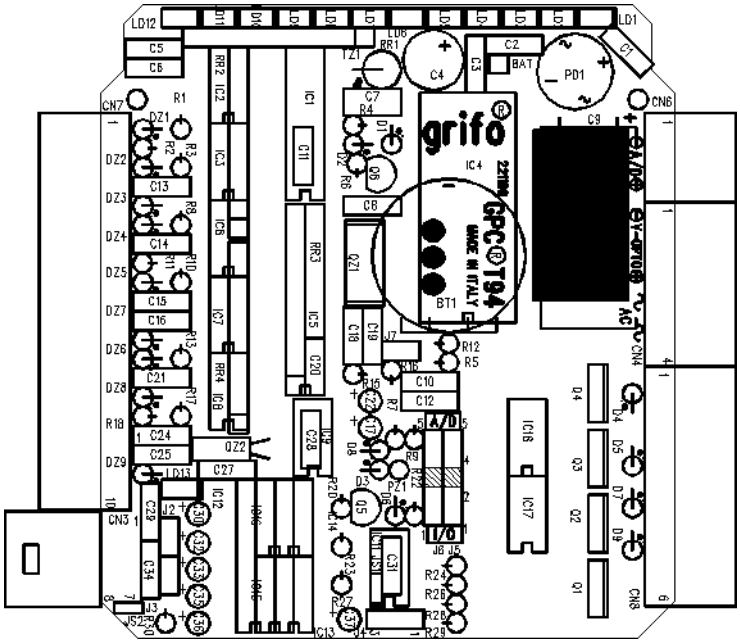


FIGURE 6: COMPONENTS MAP GPC® T94 (SOLDER SIDE)

INSTALLATION

In this chapter there are the information for a right installation and correct use of the card. The User can find the location and functions of each connector, jumper and some explanatory diagrams.

CONNECTIONS

The **GPC® R/T94** module has 5 connectors that can be linked to other devices or directly to the field, according to system requirements. In this paragraph there are connectors pin out, a short signals description (including the signals direction) and connectors location. To easily locate them please refer to figures 8 and 9, while for further informations abuot the connections please refer to the following figures.

CN4 - POWER SUPPLY CONNECTOR

CN4 is a 4 pins, quick release, screw terminal connector. CN4 is used to power the card when on **GPC® R94** or **GPC® T94** there is the switching power supply section. If switching power supply section is not present the card must be powered through CN3 connector.

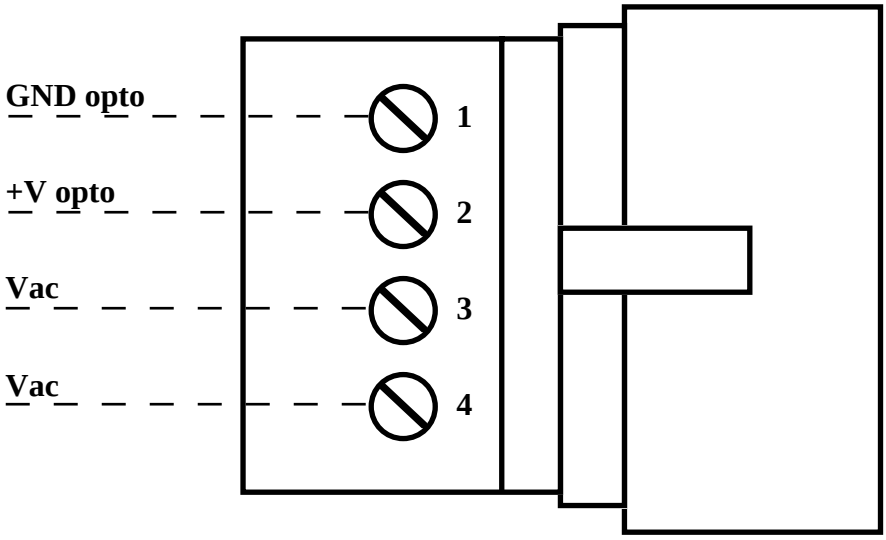


FIGURE 7: CN4 - POWER SUPPLY CONNETTOR

Signals description:

+V opto = I - +Vopto power supply.
GND opto = - Opto ground signal.
Vac = I - Low voltage power supply line +10÷40 Vdc or 10÷24Vac.
Vac = I - Low voltage power supply line +10÷40 Vdc or 10÷24Vac.

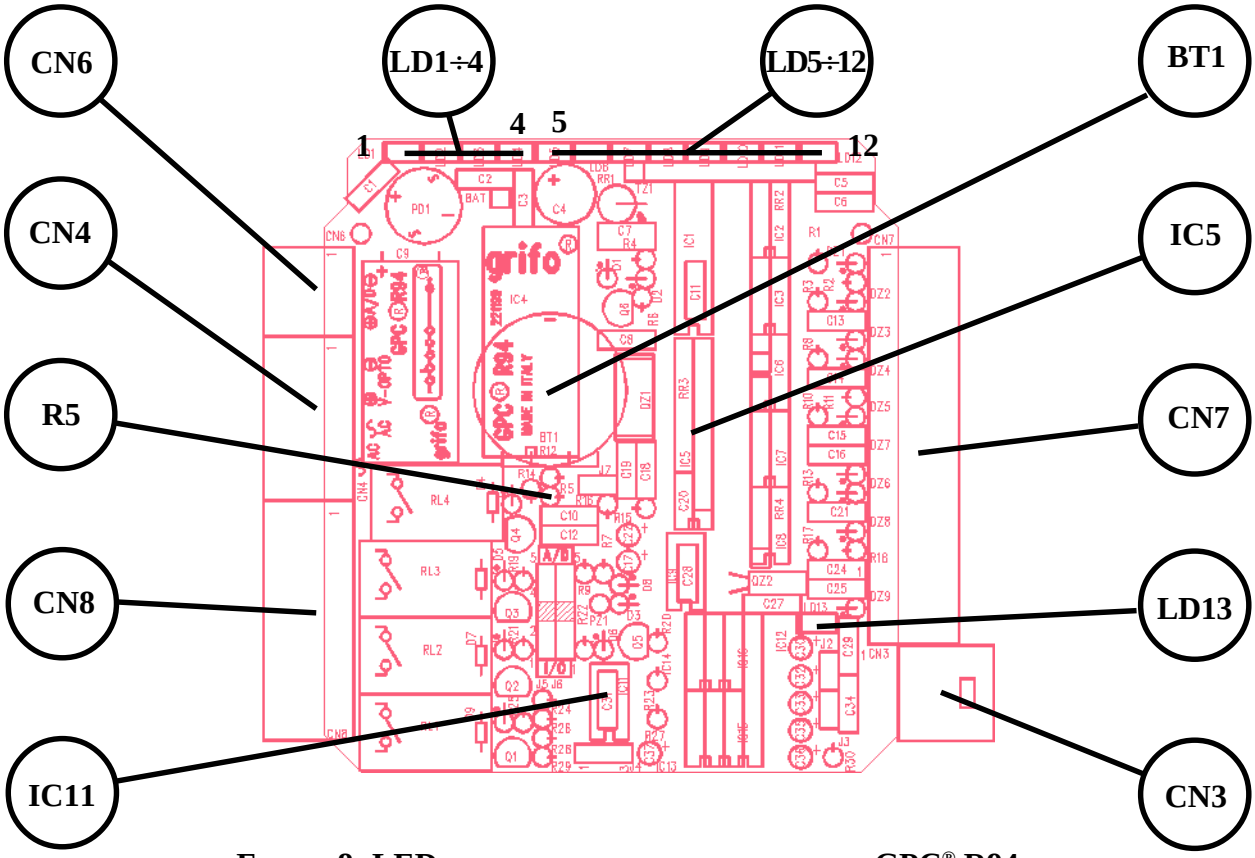


FIGURE 8: LEDS, CONNECTORS, ETC. LOCATION ON GPC® R94

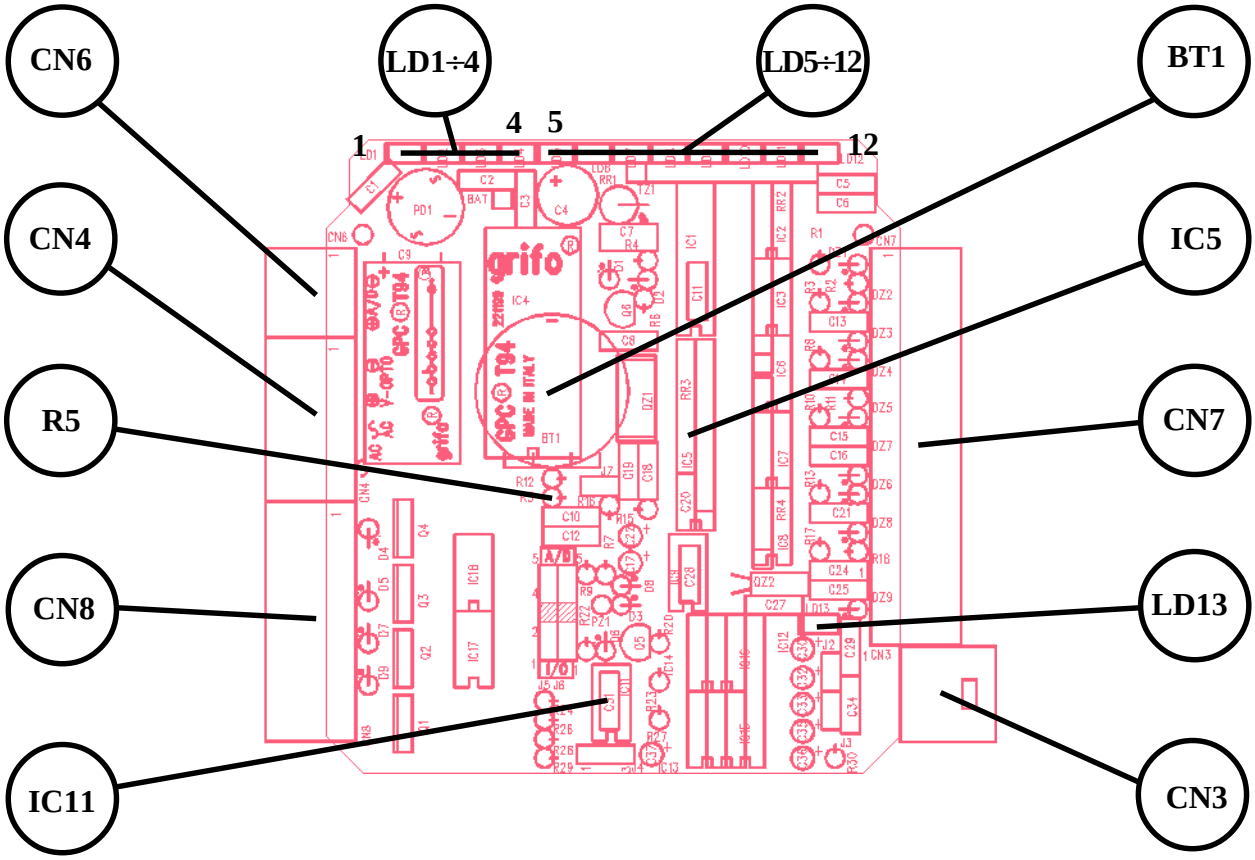


FIGURE 9: LEDS, CONNECTORS, ETC. LOCATION ON GPC® T94

CN3 - CONNECTOR FOR SERIAL LINE AND POWER SUPPLY

If the **GPC® R94** or **GPC® T94** has the switching power section mounted, trough CN3 the user can draw the two galvanically isolated voltages and can use them for external loads supply.
In the other case (switching not present) trough CN3 the user must give the two voltages (+5 Vdc and +Vopto) for card power supply. Also, on CN3 connector are available the buffered signals for TTL, RS 232, RS 422, RS 485 or Current Loop serial line communication, each of the listed electric protocols (CCITT normatives compliant). All the signals are placed in order to reduce interference and electrical noise. The female connector can be ordered directly to **grifo®** (code **CKS.AMP8**) or to AMP (female AMP 4+4 pins connector Mod II P/N: 280365; pins to be crimped as loose pieces P/N:182206-2).

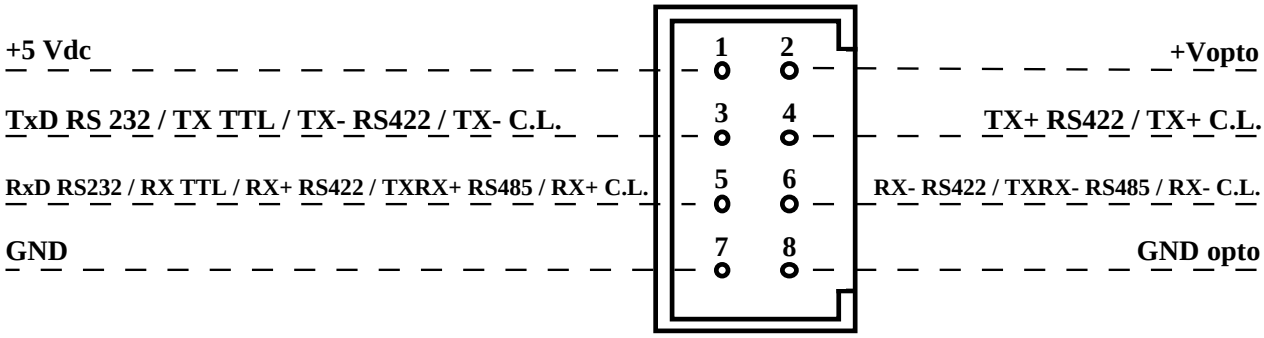


FIGURE 10: CN3 - CONNECTOR FOR SERIAL LINE AND POWER SUPPLY

Signals description:

RX TTL	=	I	- Receive Data of TTL serial line.
TX TTL	=	O	- Transmit Data of TTL serial line.
RxD RS 232	=	I	- Receive Data of RS 232 serial line.
TxD RS 232	=	O	- Transmit Data of RS 232 serial line.
RX- RS 422	=	I	- Receive Data Negative of RS 422 serial line.
RX+ RS 422	=	I	- Receive Data Positive of RS 422 serial line.
TX- RS 422	=	O	- Transmit Data Negativ of RS 422 serial line.
TX+ RS 422	=	O	- Transmit Data Positive of RS 422 serial line.
TXRX- RS 485	=	I/O	- Transmit Receive Data Negative of RS 485 serial line.
TXRX+ RS 485	=	I/O	- Transmit Receive Data Positive of RS 485 serial line.
RX- C.L.	=	I	- Receive Data Negative of Current Loop serial line.
RX+ C.L.	=	I	- Receive Data Positive of Current Loop serial line.
TX- C.L.	=	O	- Transmit Data Negative of Current Loop serial line.
TX+ C.L.	=	O	- Transmit Data Positive of Current Loop serial line.
+5 Vdc	=	I/O	- Positive terminal of +5 Vdc power supply.
GND	=		- Digital ground signal.
+V opto	=	I/O	- Positive terminal of +V opto power supply.
GND opto	=		- Ground signal for +V opto power supply.

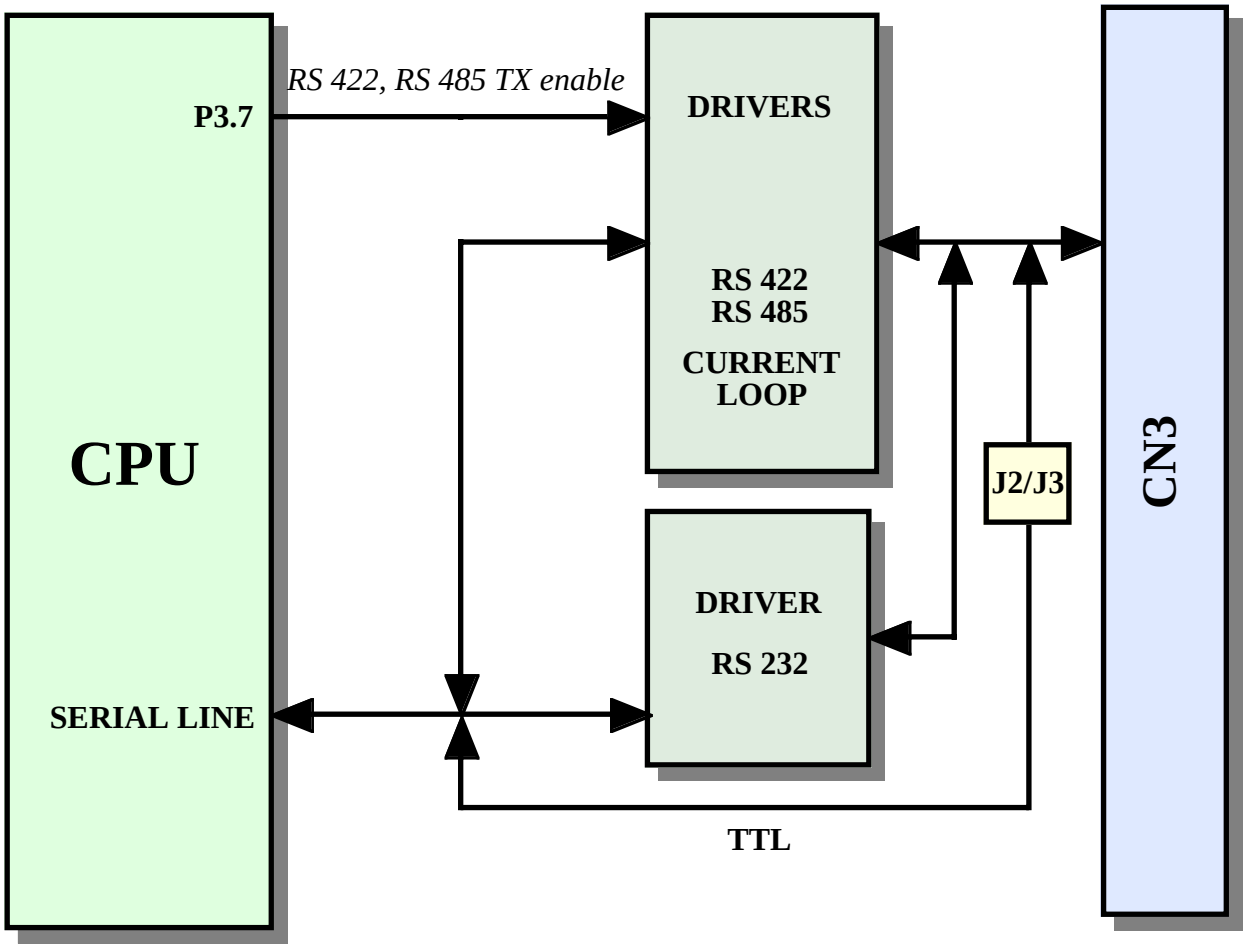


FIGURE 11: SERIAL COMMUNICATION DIAGRAM

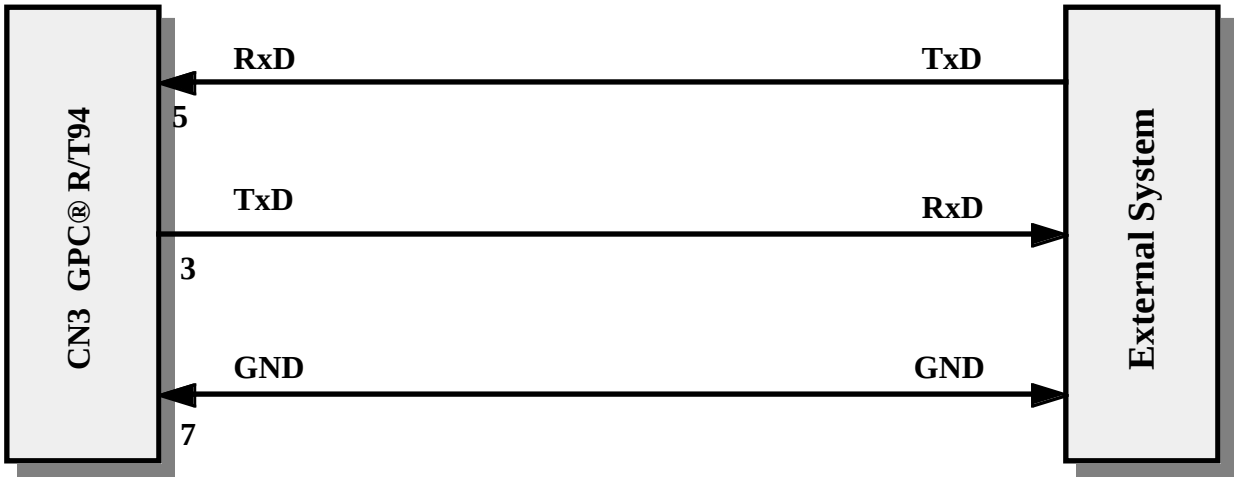


FIGURE 12: RS 232 POINT TO POINT CONNECTION EXAMPLE

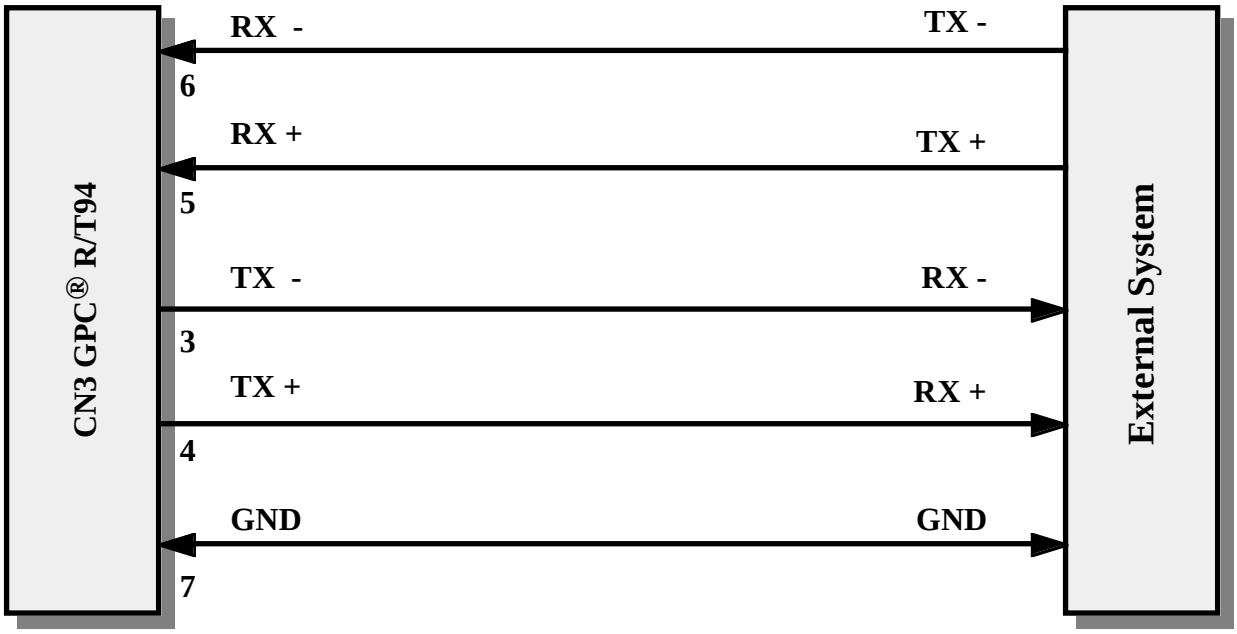


FIGURE 13: RS 422 POINT TO POINT CONNECTION EXAMPLE

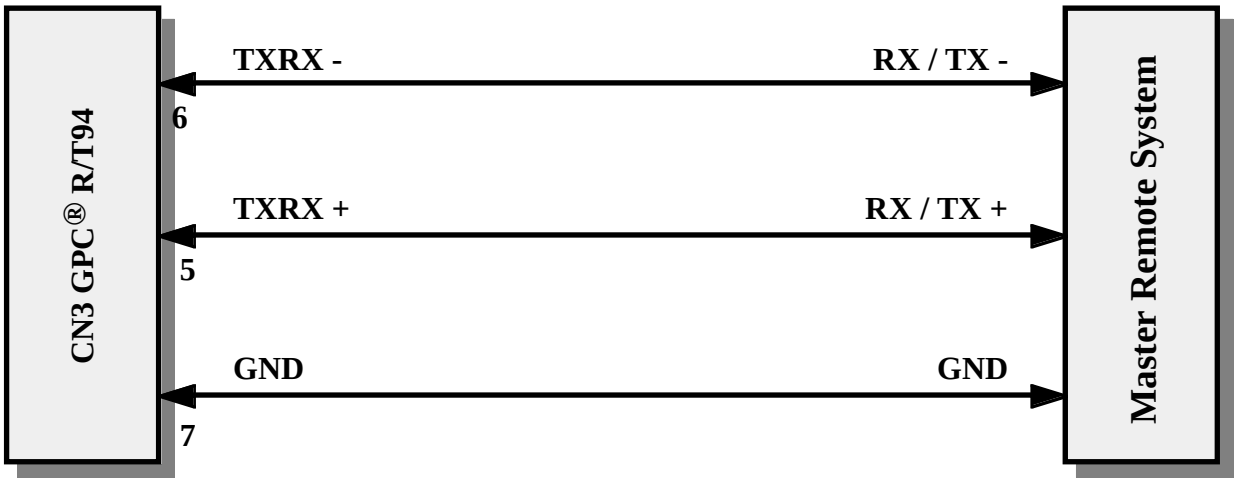


FIGURE 14: RS 485 POINT TO POINT CONNECTION EXAMPLE

NOTE

In RS 485, using firmware **ALB x94**, only 9 bit Master-Slave communication mode is possible.

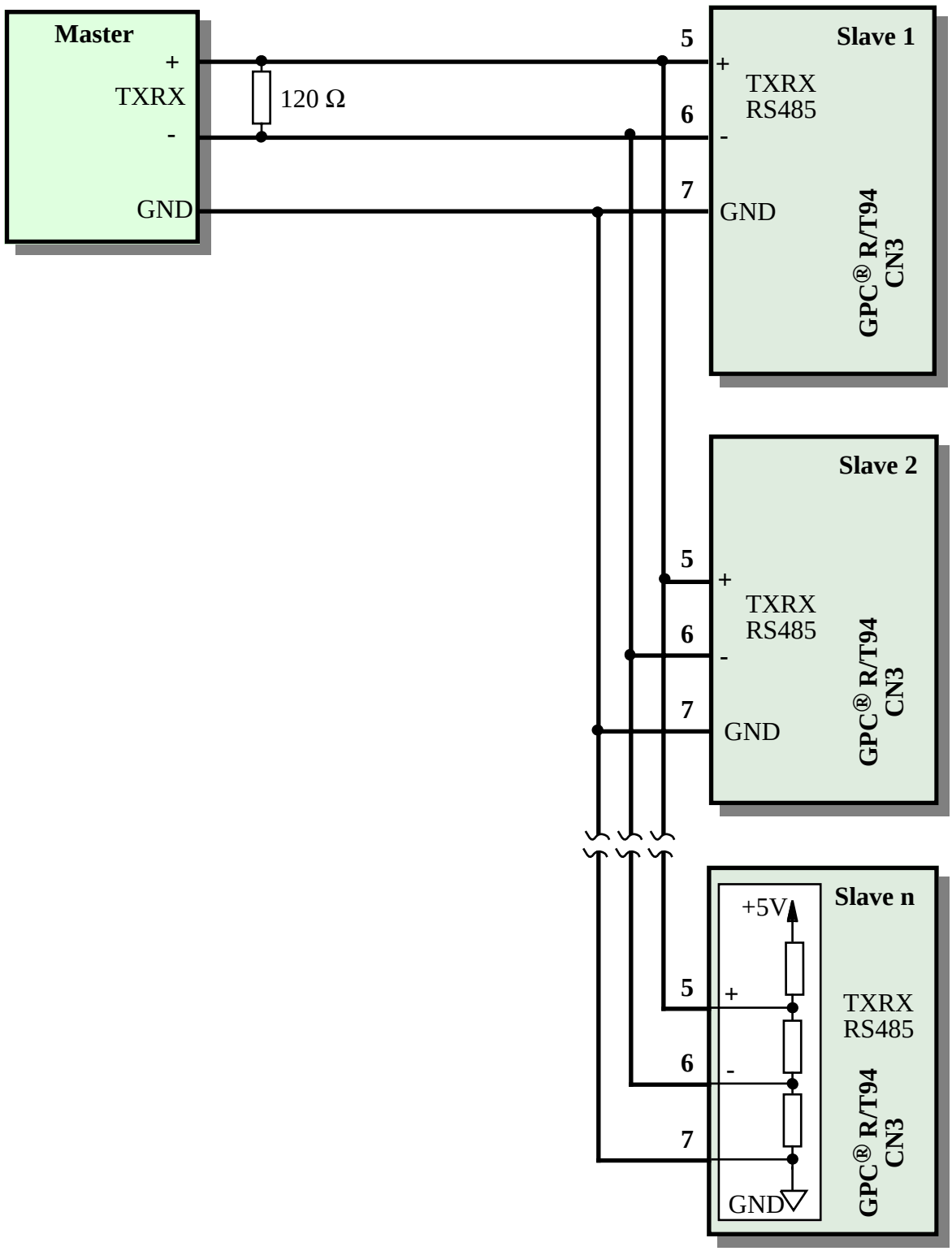


FIGURE 15: RS 485 NETWORK CONNECTION EXAMPLE

Please remark that in a RS 485 network two forcing resistors must be connected across the net and two termination resistors (120 Ω)must be placed at its extrems, respectevely near the Master unit and the Slave unit at the greatest distance from the Master.
Forcing and terminating circuitry is installed on **GPC® R/T943** board. It can be enabled or disabled through specific jumers, as explained later.
For further informations please refr to TEXAS INSTRUMENTS Data-Book , "*RS 422 and RS 485 Interface Cicuits*", the introduction about RS 422-485.

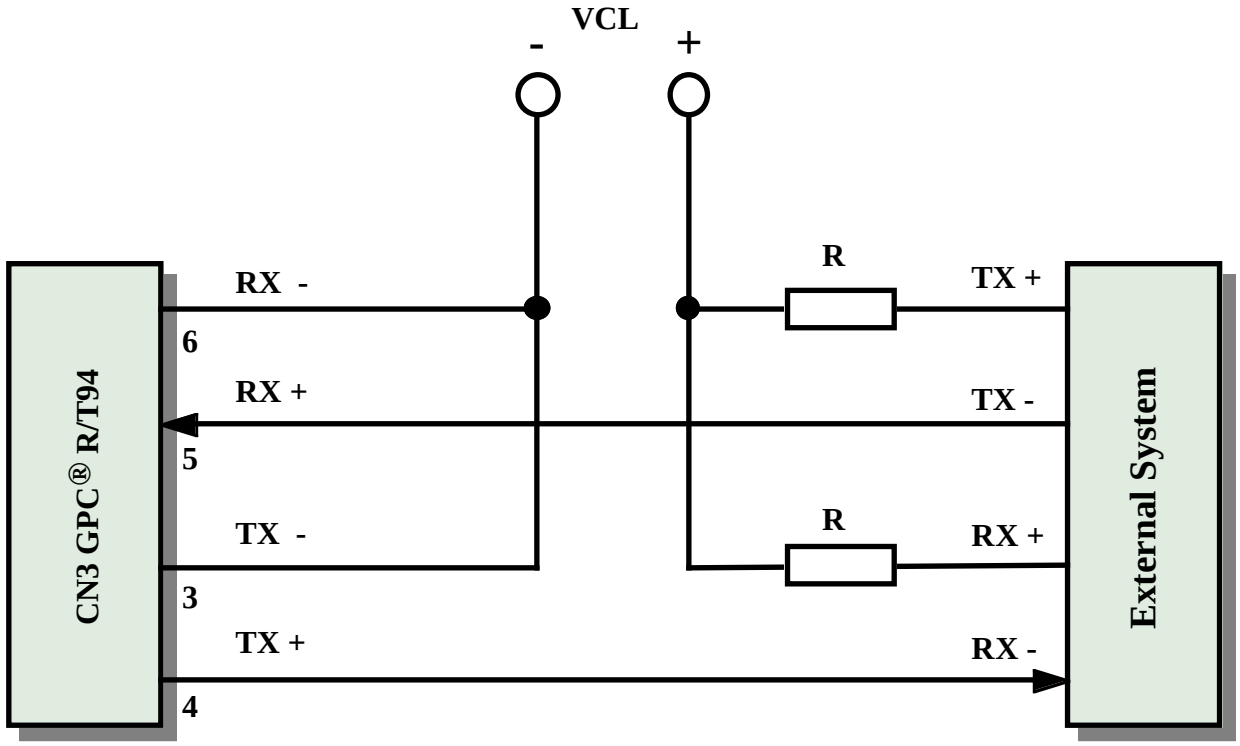


FIGURE 16: 4 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE

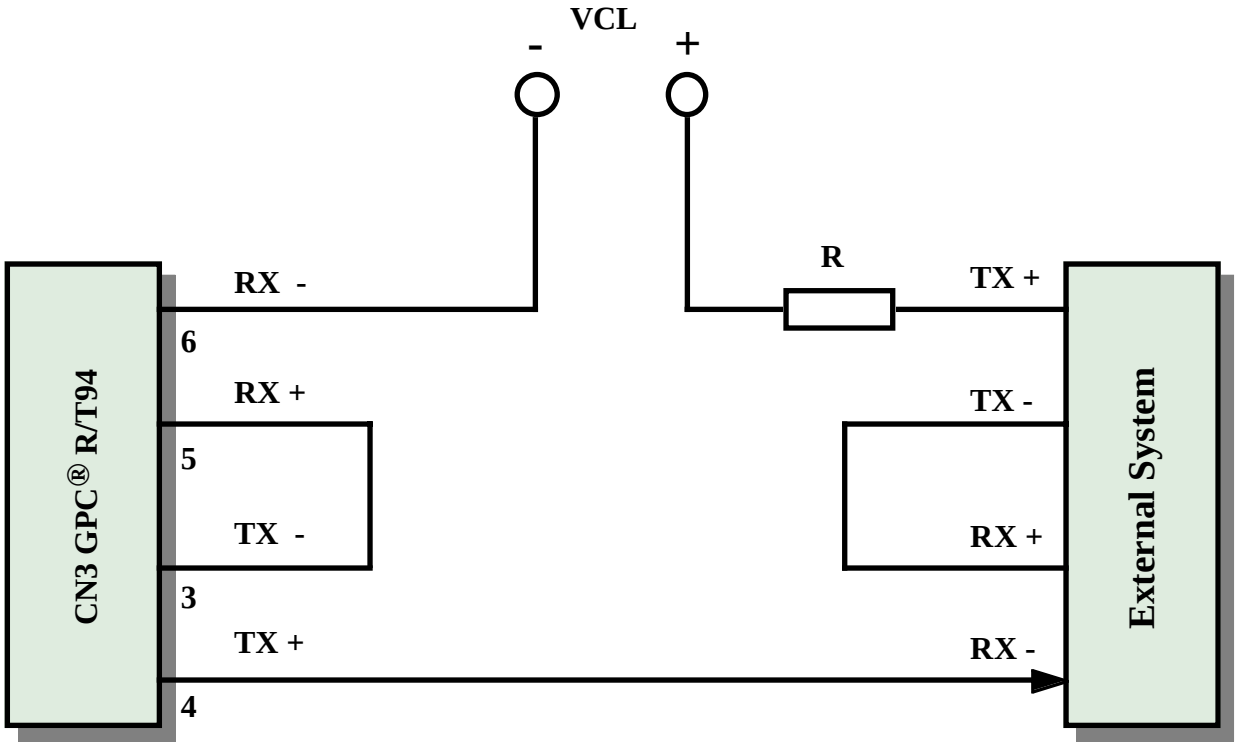


FIGURE 17: 2 WIRES CURRENT LOOP POINT TO POINT CONNECTION EXAMPLE

NOTE

In 2 wires Current Loop, using firmware **ALB x94**, only 9 bit Master-Slave communication mode

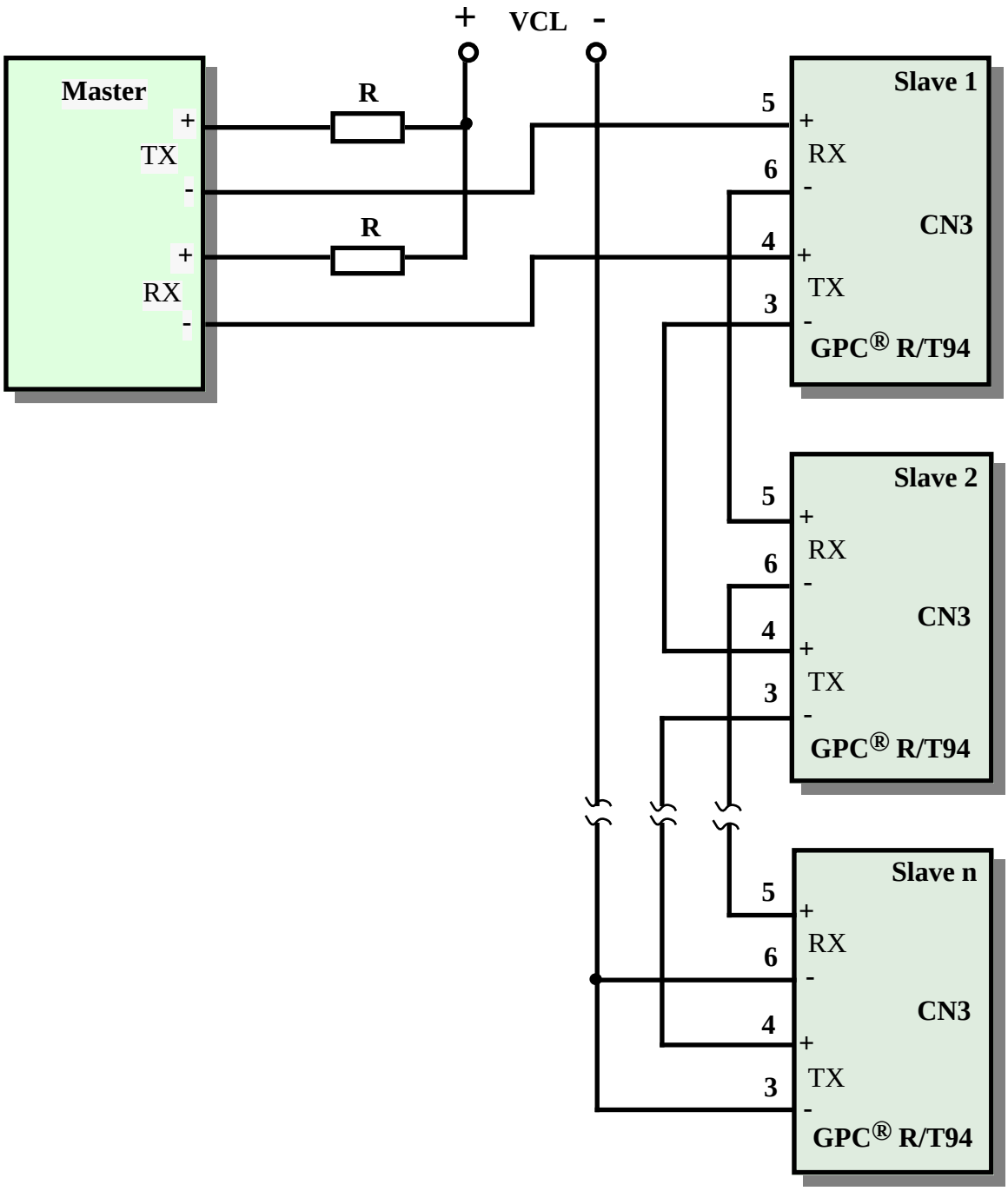


FIGURE 18: PASSIVE CURRENT LOOP NETWORK CONNECTION EXAMPLE

Possible Current Loop connections are two: 2 wires and 4 wires. These connections are shown in figures 16 and 17 where it is possible to see the voltage for **VCL** and the resistances for current limitation (**R**). The supply voltage varies in compliance with the number of connected devices and voltage drop on the connection cable.
The choice of the values for these components must be done cosidering that:
- circulation of a **20 mA** current must be guaranteed;
- potential drop on each transmitter is about **2.35 V** with a 20 mA current;
- potential drop on each receiver is about **2.52 V** cwith a 20 mA current;
- in case of shortciruit each transmitter must dissipate at most **125 mW**;
- in case of shortciruit each receiver must dissipate at most **90 mW**.

For further info please refer to HEWLETT-PACKARD Data Book, (**HCPL 4100** and **4200** devices).

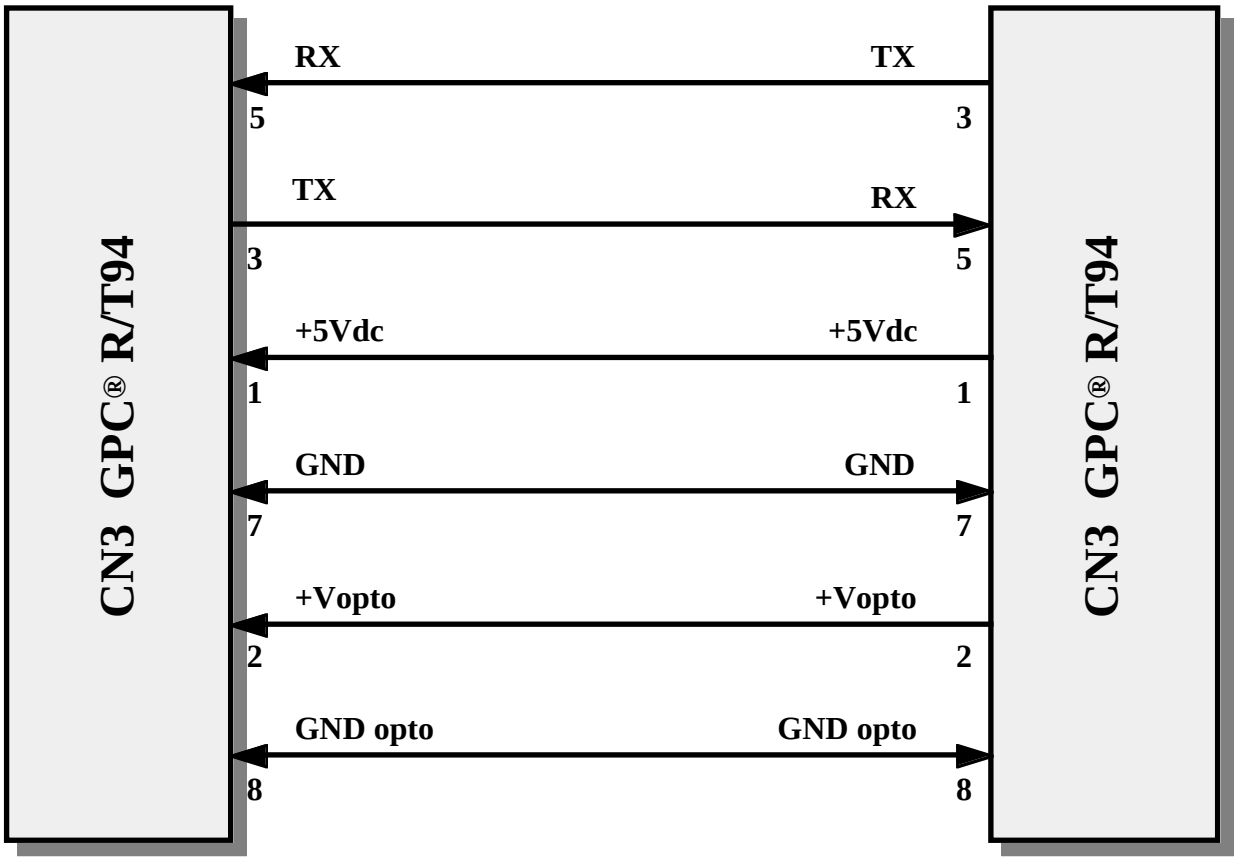


FIGURE 19: TTL SERIAL COMMUNICATION AND POWER SUPPLY CONNECTION EXAMPLE

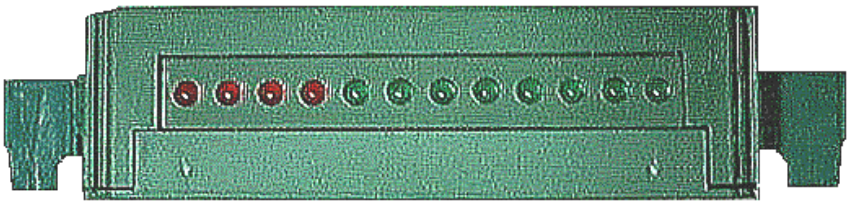


FIGURE 20: GPC® R94 CARD PHOTO

CN7 - CONNECTOR FOR OPTOCOUPLED INPUTS

CN7 is a 10 pins, quick release, screw terminal connector. CN7 is used to connect the 9 optocoupled NPN input signals that the card manages. 8 inputs are connected to the 74HCT166 while the AUX input is connected to the CPU. On the connector are available the open collector inputs and the opto ground signal.

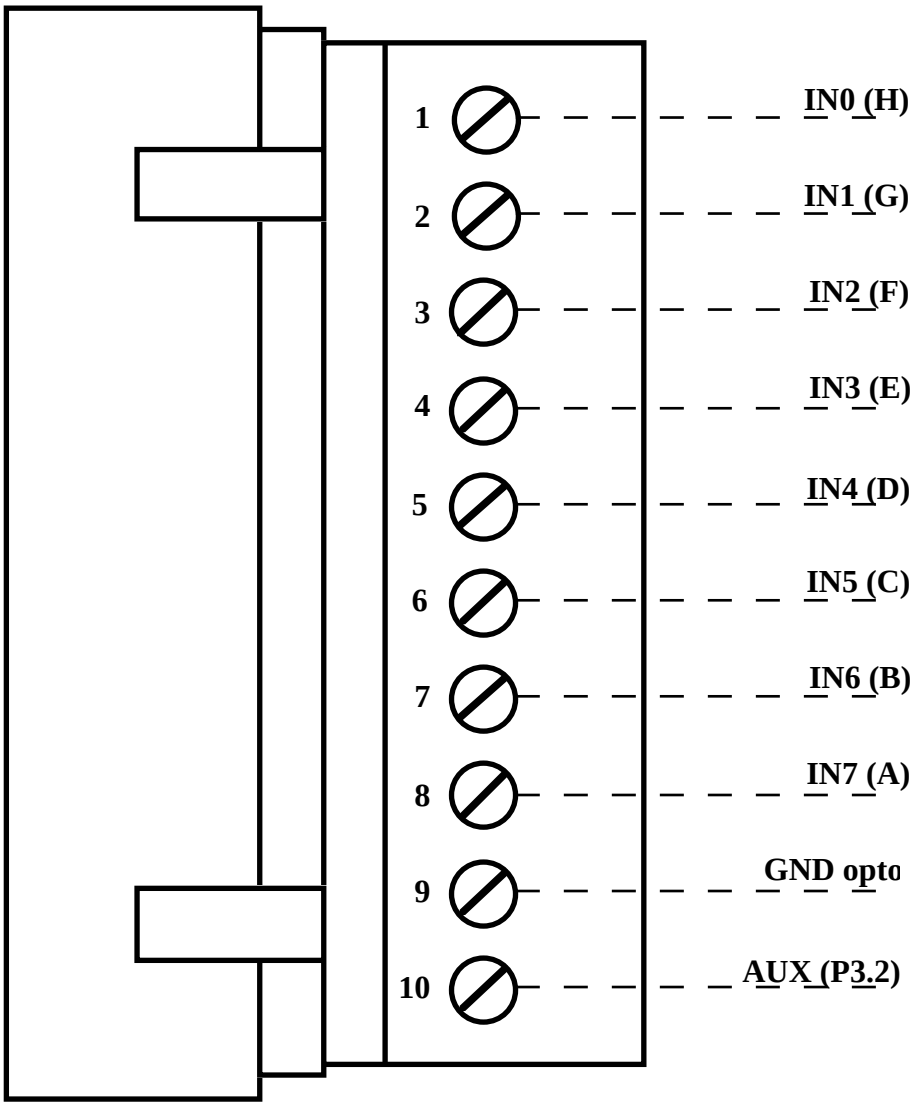


FIGURE 21: CN8 - CONNECTOR FOR OPTOCOUPLED INPUTS

Signals description:

IN0÷IN7 = I - Open collector NPN input, connected to 74HCT166 (range H÷A).
AUX = I - Open collector NPN input, connected to CPU P3.2.
GND opto = - Opto ground signal.

The inputs available on the board are optocoupled, each one driving a LED for visual feed-back (the LED wil light when the input will be connected to the ground signal). The inputs are suitable for **NPN** type drivers, to connect them to **PNP** drivers yuo should interpose a **PBI 01** module.
The 9 inputs section is shown in the diagram below.
Optocouplers power supply may be granted by connector CN4.

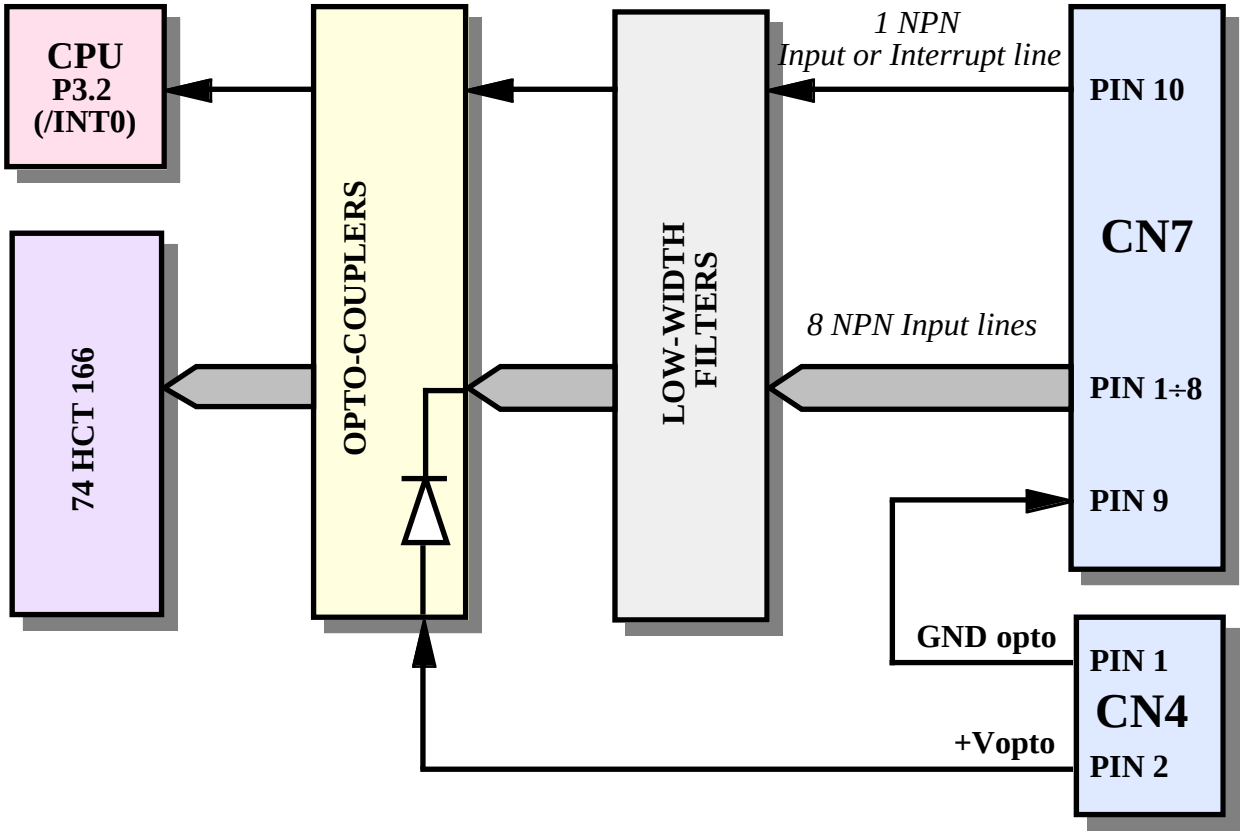


FIGURE 22: DIGITAL INPUTS CONNECTION DIAGRAM

CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR

CN8 is a 6 pins, quick release, screw terminal connector. On CN8 are available the N.O. 4 relays outputs and their relative commons. The maximum external load for each line is **5 A** with a maximum tension or 30 **Vdc** or 250 **Vac**. The outputs are managed through a set of microprocessor I/O pins, opportunaly buffered, carefully selected to simplify the software management (for further informations please refer to the chapter “PERIPHERAL DEVICES SOFTWARE DESCRIPTION”

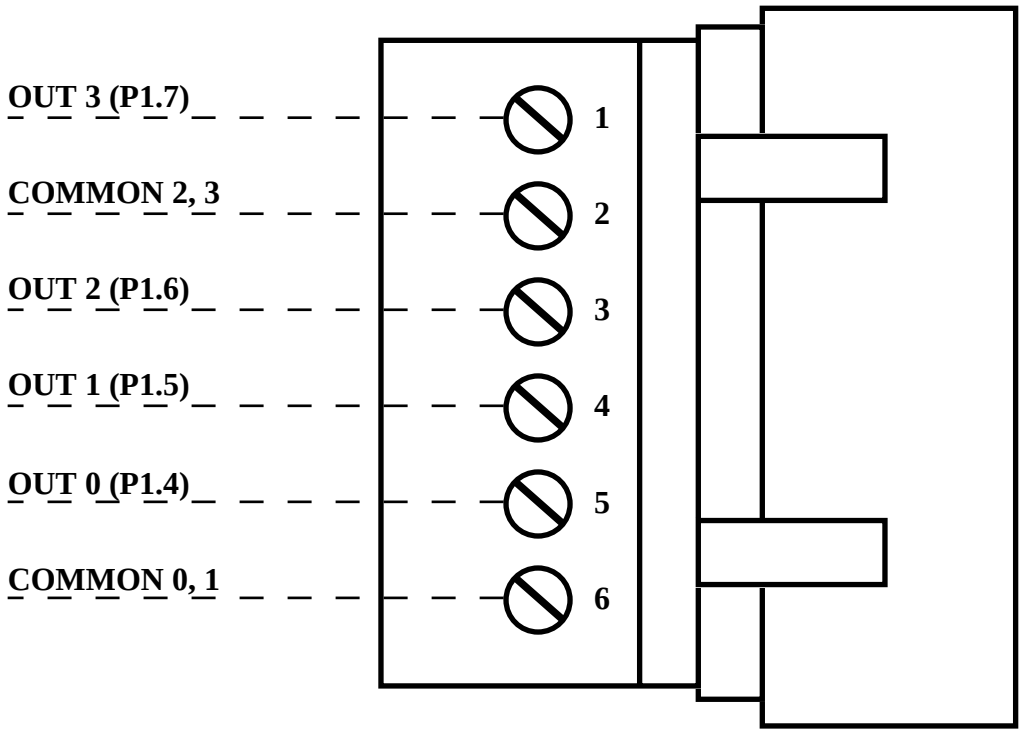


FIGURA 23: CN8 - GPC® R94 RELAY OUTPUTS CONNECTOR

Legenda:

OUT n (Px.y) = O - N.O. contact of relay n, driven by the pin Px.y of the microcontroller;
COMMON n, m = - Common contact of relays n and m.

Each relay output signal is connected to a LED whose purpose is visual feed-back (the LED will light whenever the relay contact is closed). The relays are driven by 4 PNP transistors that are managed through microcontrolles's I/O pins.

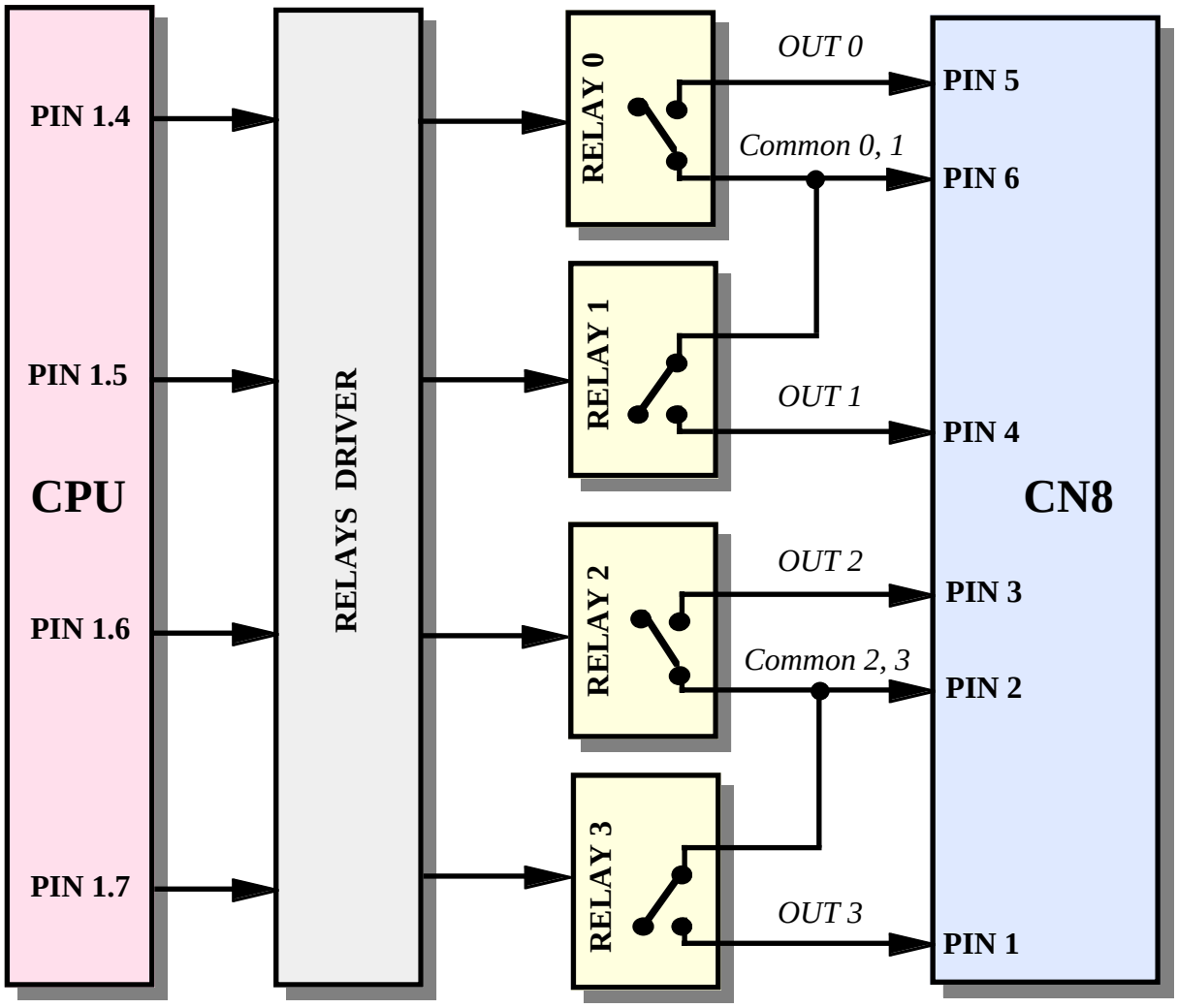


FIGURA 24: RELAY OUTPUTS DIAGRAM

CN8 - GPC® T94 TRANSISTOR OUTPUTS CONNECTOR

CN8 is a 6 pins, quick release, screw terminal connector. On CN8 are available the 4 open collector Darlington NPN transistor outputs and their relative commons (emitter). The maximum external load for each line is **4 A** not continuative with 45 Vdc. The transistors, being without heat sink,can drive in continuative way a resistive load that absorbs at most **600 mA** with a maximum voltage if **45 Vdc** only if the work temperature is 20° C. All the signals are provided with a back EMF protection diode which suppresses eventual inductive voltages when loads like power relays, solenoids etc. are driven. In this case the load supply must be connected to the +VL signal. The outputs are managed through a set of microprocessor I/O pins, opportunaly buffered, carefully selected to simplify the software management (for further informations please refer to the chapter "PERIPHERAL DEVICES SOFTWARE DESCRIPTION".

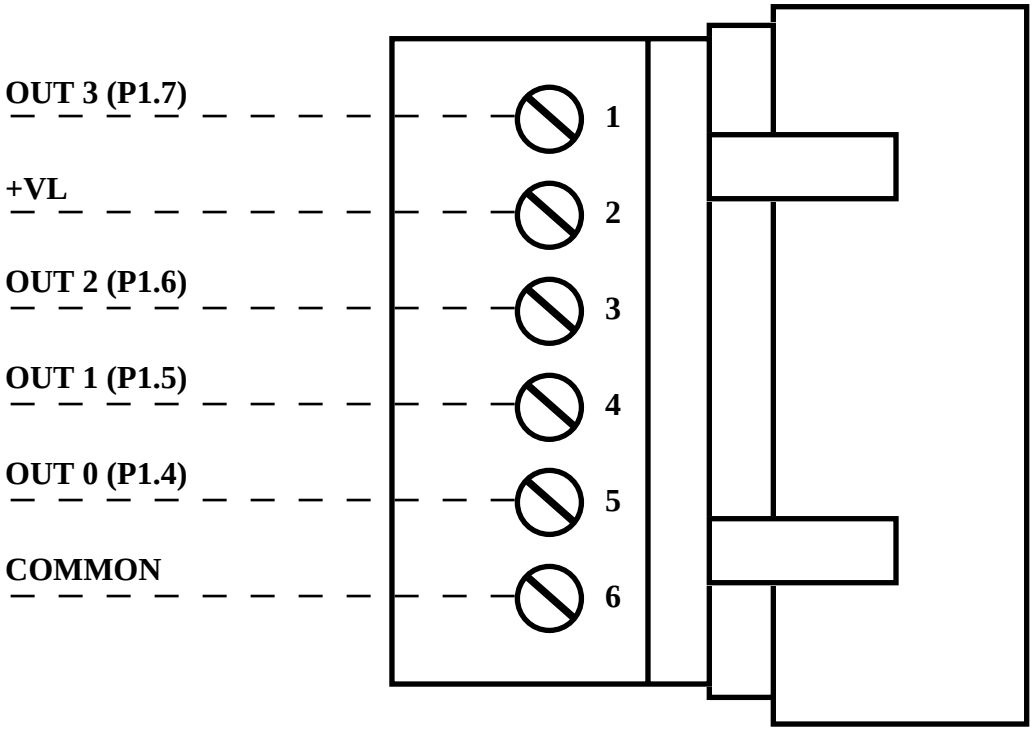


FIGURE 25: CN8 - GPC® T94 TRANSISTOR OUTPUTS CONNECTOR

Signals description:

OUT n (Px.y) = O -Open collector contact of n-th NPN transistor, driven by microcontroller's pin Px.y.
COMMON = - Common emitter of transistors.
+VL = I - Back EMF protection diodes power supply (power supply of load).

Each output is optocoupled and it has a own LED that shows the line status (the LED is turned ON when the transistor is in conduction).
The final stage of such outputs is made of an NPN Darlington transistor in Open Collector, provided with back EMF protection diode, all the output's emitters are connected to an unique common point.

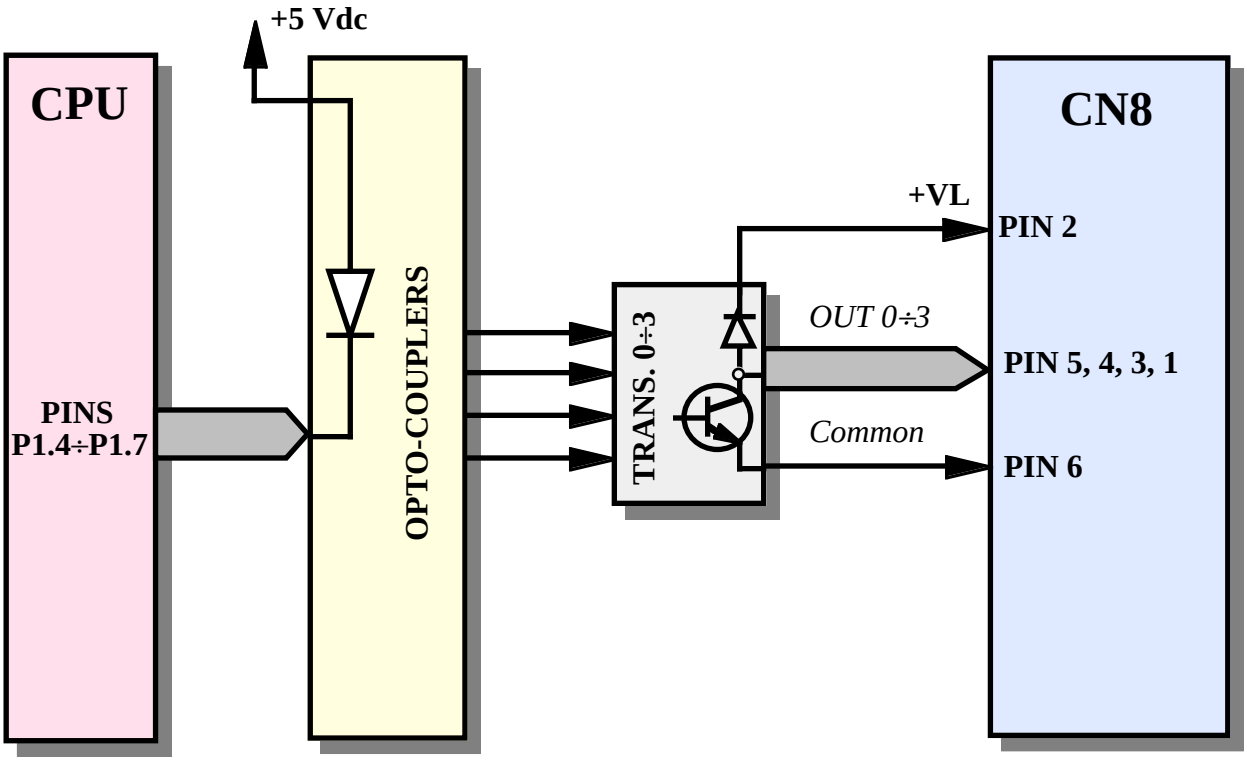


FIGURA 26:TRANSISTOR OUTPUTS DIAGRAM

CN6 - A/D CONVERTER INPUT CONNECTOR AND TTL I/O LINES

CN6 is a 2 pins, quick release, screw terminal connector used to connect the two multi-purpose signals (A/D Converter input, I/O TTL or analog comparator inputs) of **GPC® R/T94**.
The on board circuitry is configurable through jumers J5 and J6, as explained later, to solve the User problematics without further expenses.
If th A/D conversion is selected, please remark that a filtering capacitors is connected and that the input signal range must be 0÷10 V or 0÷20 mA or 4÷20 mA, while the conversion is managed through the microprocessor's analog comarator registers, as decribed in the chapter“PERIPHERAL DEVICES SOFTWARE DESCRIPTION”.

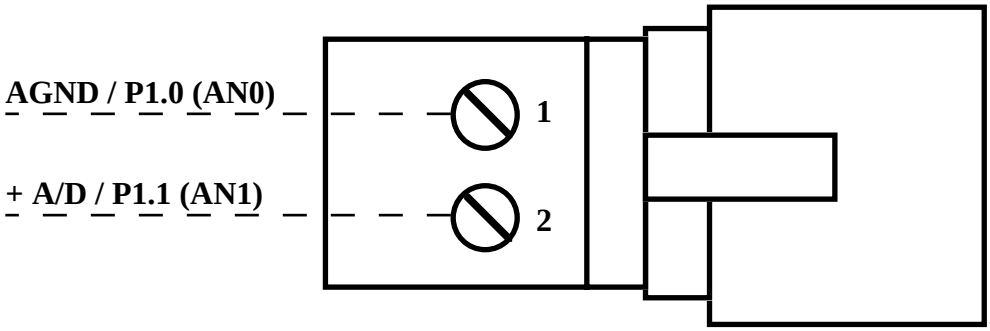


FIGURE 27: CN6 - A/D CONVERTER INPUT CONNECTOR

Signals description:

Px.y (ANy)	= I/O	- I/O TTL signal connected to microcontroller's pin Px.y or input ANy of the analog comparator.
+ A/D	= I	- A/D converter circuitry input signal.
AGND	=	- Analog ground signal.

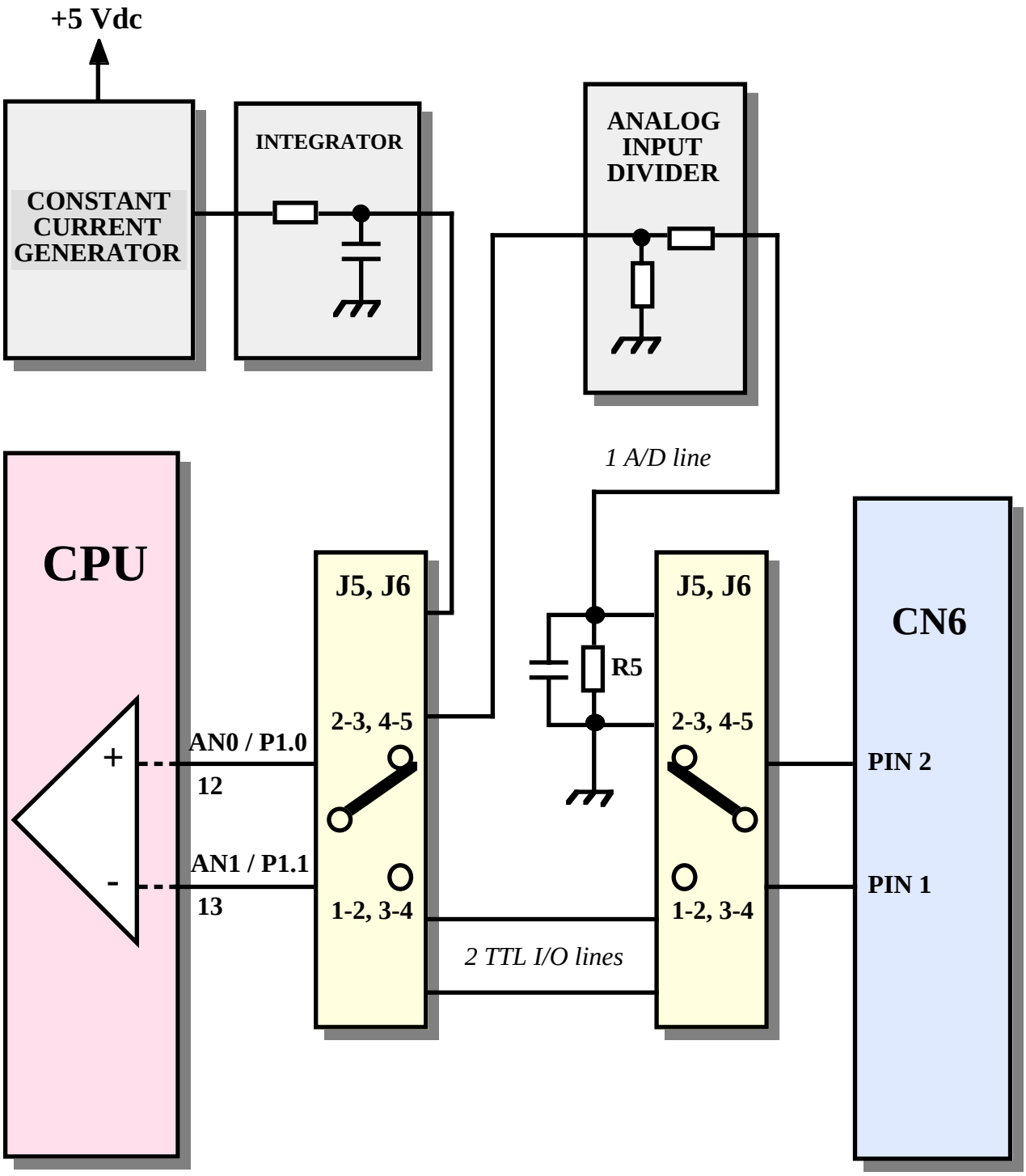


FIGURA 28: A/D CONVERTER INPUT DIAGRAM

LEDS

On **GPC®R/T 94** boards there are 15 LEDs that shows the card status, as described in the following table:

LEDs	COLOUR	PURPOSE
LD1÷LD4	Red	They indicate the status of digital outputs 0÷3. The LED is on when the output OUTn is connected to COMMON n.
LD5÷LD12	Green	They indicate the status of digital input 0÷7. The LED is on when theinput INPn is connected to GND opto.
LD13	Green	It indicates the status of AUX signal. The LED is on when theinput AUX is connected to GND opto.

FIGURE 29: LEDS TABLE

LEDs LD1÷LD12 are visible on one edge of the card, while LED LD13 is visible only removing the cover of the container.
The main purpose of LEDs is to inform the User about card status, with a simple visual indication and in addition to this, LEDs make easier the debug and test operations of the complete system. To recognize the LED location on the card, please refer to figures 8 and 9.

INTERRUPTS

A remarkable feature of **GPC® R/T 94** cards is the powerful interrupt management. Here follows a short description of which devices can geneate interrupts and their modalities; for further informations about interrputs management please refer to the microprocessor data sheet or to the appendix B of this manual.

- AUX input -> Generates an interrupt on microcontrollor's pin /INT0 (P3.2).

- CPU peripherals -> Generate an internal interrupt. In particular the possible internal interrupt sources are the sections: Timer/Counter and serial line.

An interrupt manager that allows to activate, deactivate and mask the interrupt sources is installed on the board. It can also manage contemporary interrupts. This way the User can always react promptly and efficently to any external event, being also able to decide a priority for thr several intettupt sources.

BACK UP

On **GPC® R/T 94** is mounted a Lithium battery that keeps data on SRAM+RTC also when power supply fails. With jumper J7 the User can connect or disconnect the Back Up whenever it is needed, so the battery life time is increased. To easily locate the battery please refer to figures 8 and 9.

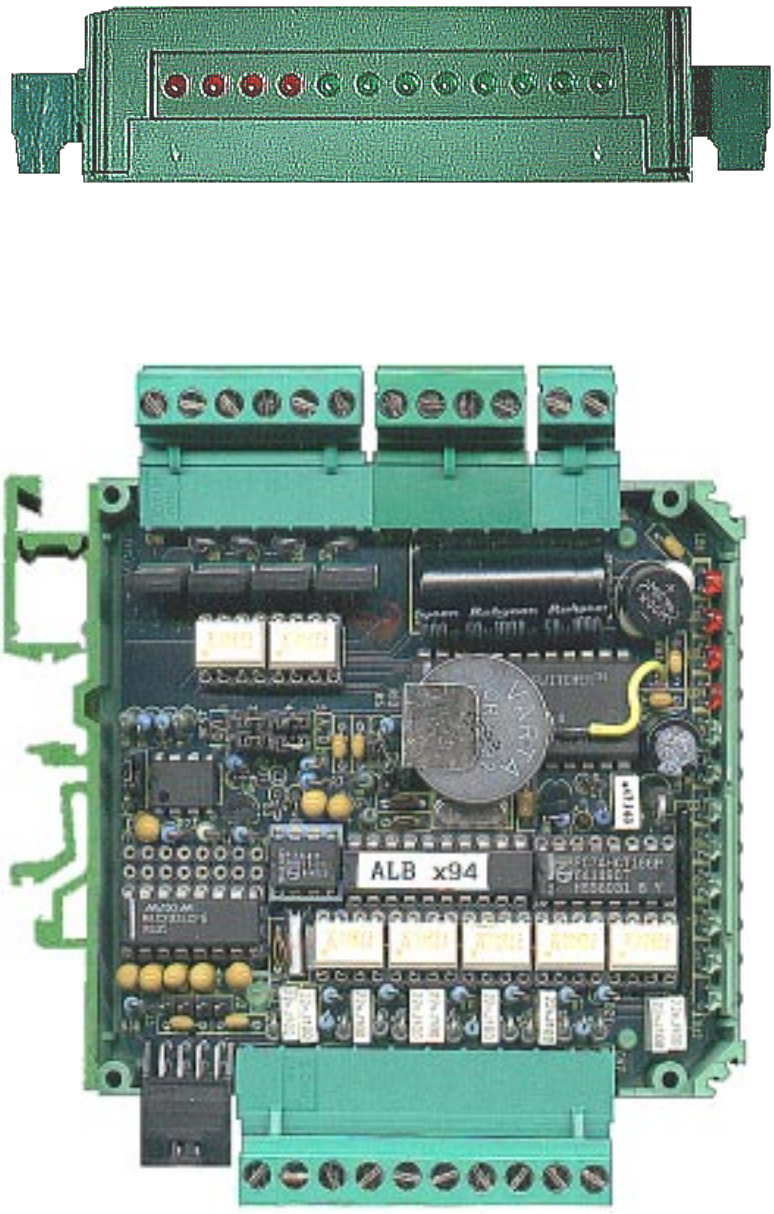


FIGURE 30: GPC® T94 CARD PHOTO

I/O CONNECTION

To prevent possible connecting problems between **GPC® R/T 94** and the external systems, the User has to read carefully the previous paragraph information and he must follow these instrunctions:

- For RS 232, RS 422, RS 485 and Current Loop signals the User must follow the standard rules of each one of these protocols;

- For all TTL signals the User must follow the rules of this electric standard. The connected digital signal must be always referred to card digital ground and if an electric insulation is necessary, then an opto coupled interface must be connected. For TTL signals, the 0V level corresponds to logic state 0, while 5V level corrisponds to logic state 1.

- The analog input (A/D Converter section) must be connected to signals within the ranges: 0÷10 V, 0÷20 mA o 4÷20mA according to card configuration. Please remark that such input, available on connector CN6, is provided with a low width filtering capacitor that warrants a greater stability of the signal acquired

- To the optocoupled input signals only the contacts to be acquired must be connected. Such contacts (relays, swithces; etc.) must connect or must not connect the input signal INx to GND opto. For logic signals correspondace, the open contact corresponds to logic state **1**, while the closed contact corresponds to logic state **0**. Correspondance is compliant to the NPN normative.

- The relays output signals, available only on **GPC® R94**, must be connected directly to the load to be driven (power relays, etc.). The boards provides the normally open contact, for which maximum current is 5A with a tension that can be as high as **30 Vdc** or **250 Vac**.
  To give the chance to drive different loads with different supplies, two different COMMON contacts are available, that connect two different groups of relays.

- The Darlington NPN output signals, available only on **GPC® T94**, must be connected directly to the load to be driven (power relays, etc.). The boards provides the Open Collector output line , for which maximum current is **4A not continuative** with a tension that can be as high as **+45 Vdc**.
  The transistors, being without heat sink,can drive in continuative way a resistive load that absorbs at most **600 mA** with a maximum voltage as high as **45 Vdc** only if the work temperature is 20° C.

ANALOG INPUTS SELECTION

One of the **GPC® R/T 94** particular features is the possibility to acquire tension and/or current signals for the A/D inpus. The CURRENT signals selection is obtained throuhg proper resistror mounted on R5 position (option **.8420**).
If the resistor is not mounted (default) the channel can acquire a TENSION signal in the range 0÷10 Vdc (default), instead if the resistor is mounted the channel can acquire a CURRENT signal.
The resistors value for the CURRENT/VOLTAGE converter section is calculated with the following formula:

R = 10 V / Imax

Normaly the precision CURRENT/VOLTAGE resistor value is **499Ω** suitable for 0÷20 mA analog inputs. For eventuel requirements outdise these standard ranges please contact **grifo®**.
Please refer to figures 8 and 9 for the resistor location.

POWER SUPPLY SECTION

One of the most important features of **GPC® R/T94** is its on board power supply circuitery; the card can be powered in two different ways to let the User solve easier the problem to supply the board in any working condition:

Without switching power supply (default):

+V opto: Supplies the optocouplers of the board's input section; it must be +24 Vdc and must be provided through pins 1 and 2 of CN4 or pins 2 and 8 of CN3.

+5 Vdc: Supplies the control logic and the board's output section: it must be +5 Vdc ± 5% and must be provided through pins 1 and 7 of CN3.

*With switching power supply (option **.SW**):*

+V opto: Supplies the optocouplers of the board's input section; it must be +24 Vdc and must be provided through pins 1 and 2 of CN4 or pins 2 and 8 of CN3.

Vac: Supplies the control logic and the board's output section through the on board switching power supply: it must be 10÷40 Vdc or 8÷24 Vac (polarity is irrilevant) and must be provided through pins 3 and 4 of CN4. The power supply circuit generates the necessary voltages for the card, starting from all the standard industrial source like mains, power transformer, battery, solar cell, etc. It is possible to supply external loads with the +5 Vdc tension available on pins 1 and 7 of CN3.
Please remark that the on board switching supply section is provided with its own diode bridge, so if the supply is continuous tension the ground digital signal (GND) doesn't have the same potential of the ground signal available on CN4.

To warrant the maximum electrical noise immunity and so the correct board's functions, power supply voltages must be galvanically isolated; to assure this condition even starting from mains supply the power supply **EXPS-2** can be ordered.
The power supply type must be specified at the moment of the order and can be installed only by **grifo®** technicians. The power supply circuit was designed for reducing the consumption (the microprocessor power down and idle mode is available) and for increasing the electrical noise immunity. On board there is a protection circuit aganist voltage peaks by **TransZorb™**.

JUMPERS

On **GPC® R/T94** there are 8 jumpers, two of them are solder jumpers, for card configuration. Connecting these jumpers, the user can define for example the memory type and size, the peripheral devices functionality and so on. Below there is the jumpers list, location and function:

JUMPERS	N. PINS	PURPOSE
J2, J3	2	They configure the serial line electric protocol.
J4	3	Selects directionality and activation modality for RS 422, RS 485 serial line.
J5, J6	5	They configure the two signals on connector CN6 as analog A/D input or general purpose I/O TTL.
J7	2	Connects the on board Lithium battery to the Back Up circuitry.
JS1, JS2	2	They connect the RS 433-RS 485 forcing and terminating circuitry.

FIGURE 31: JUMPERS SUMMARIZING TABLE

The following tables describe all the right connections of **GPC® R/T94** jumpers with their relative functions. To recognize these valid connections, please refer to the board printed diagram (serigraph) or to figures 3÷6 of this manual, where the pins numeration is listed; for recognizing jumpers location, please refer to figures 33 and 36. The "*" used in the following tables, denotes the default connection, or on the other hand the connection set up at the end of testing phase, that is the configuration the user receives.

3 AND 5 PINS JUMPERS

JUMPERS	CONNECTION	PURPOSE	DEF.
J4	position 1-2	Selects the communication for the serial line as RS 485 (2 wires half duplex).	
	position 2-3	Selects the communication for the serial line as RS 422 (4 wires full duplex or half duplex).	*
J5, J6	position 1-2, 3-4	Configure pins of CN6 as 2 general purpose I/O TLL signals.	
	position 2-3, 4-5	Configure pins of CN6 as A/D conversion analog signal.	*
	position 3-3	Connects microprocessor pin P1.1 (AN1) to microprocessor pin P1.0 (AN0).	

FIGURE 32: 3 AND 5 PINS JUMPERS TABLE

Connection in position 3-3 for jumpers J5 and J6 means to connect the pins number 3 of both the jumpers.

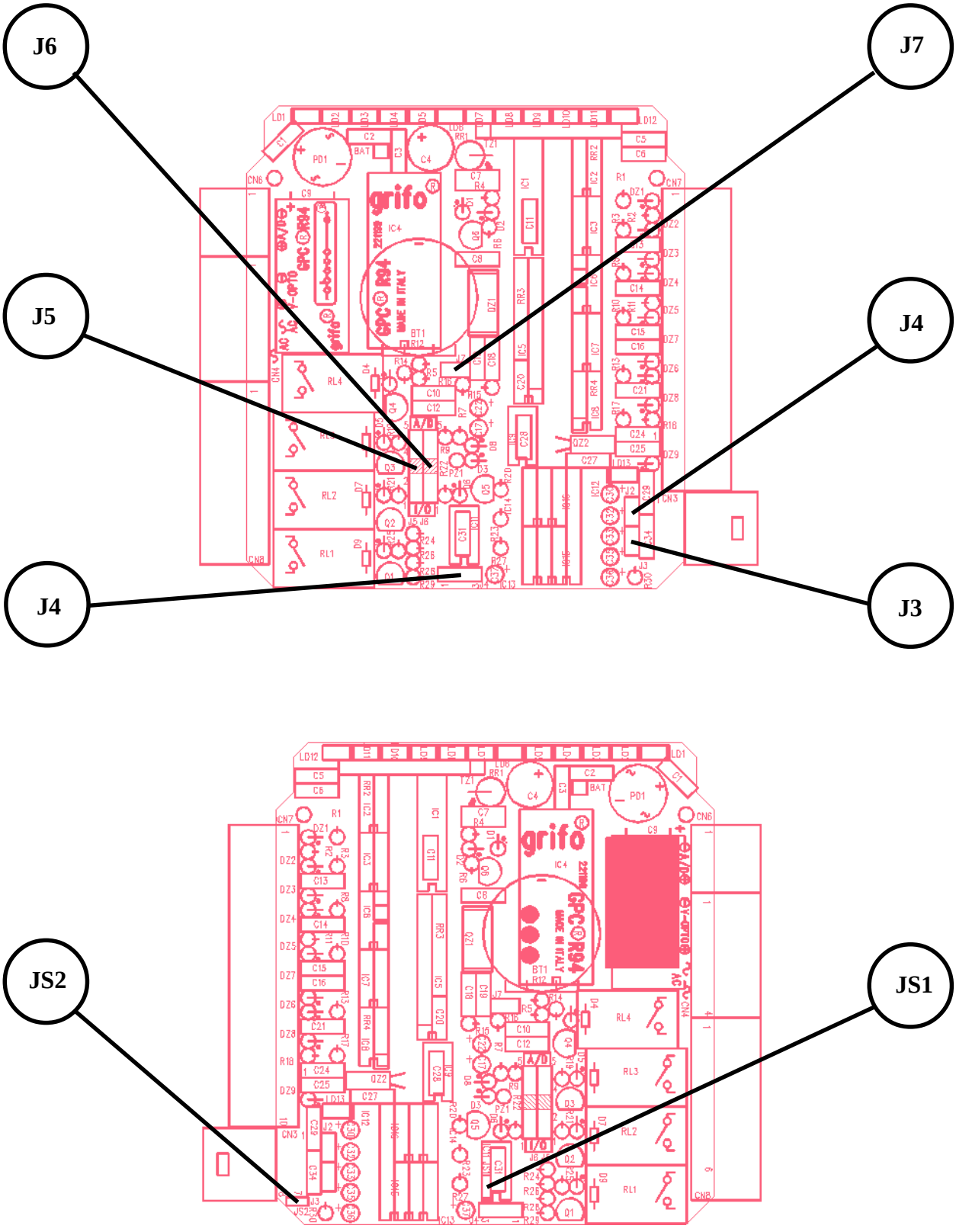


FIGURE 33: GPC® R94 JUMPERS LOCATION

2 PINS JUMPERS

JUMPERS	CONNECTION	PURPOSE	DEF.
J2, J3	not connected	They configure the serial line in RS232, RS422, RS485 o Current Loop (please see paragraph "SERIAL COMMUNICATION"for further informations).	*
	connected	They configure the serial line in TTL (please see paragraph "SERIAL COMMUNICATION"for further informations).	
J7	not connected	Does not connect Lithium battery BT1 to the Back Up circuitry.	*
	connected	Connects Lithium battery BT1 to the Back Up circuitry.	
JS1, JS2	not connected	Do not connect the termination and forcing circuitry to the RS 422 reception line or RS 485 serial line.	*
	connected	Connect the termination and forcing circuitry to the RS 422 reception line or RS 485 serial line.	

FIGURE 34: 2 PINS JUMPERS TABLE

The "*" denotes the default connection,which is set up at the end of testing phase.

MEMORY SELECTION

Le **GPC® R/T94** can manage up to 5504 bytes of memory in several configurations. For the details of the possible configurations please refer to the following table:

IC	DEVICE	SIZE
5	AT89C2051	2K Bytes (FLASH EPROM) 128 bytes (SRAM)
	AT89C4051	4K Bytes (FLASH EPROM) 128 bytes (SRAM)
9	SRAM+RTC	256 Bytes
11	EEPROM	512÷1K Bytes

FIGURE 35: MEMORY SELECTION TABLE

GPC® R/T94 is delivered in its default configuration with microprocessor AT89C4051, SRAM+RTC device installed and 512 bytes of EEPROM; every different configuration can be performed by the USer in autonomy (except for EEPROM IC11) or required in ordering phase. Below are reported the codes for the memory options available:

.EE08 -> 1K serial EEPROM

For further informations and prices of the opions please contact **grifo®**,while to easily locate memory devices please refer to figures 8 and 9.

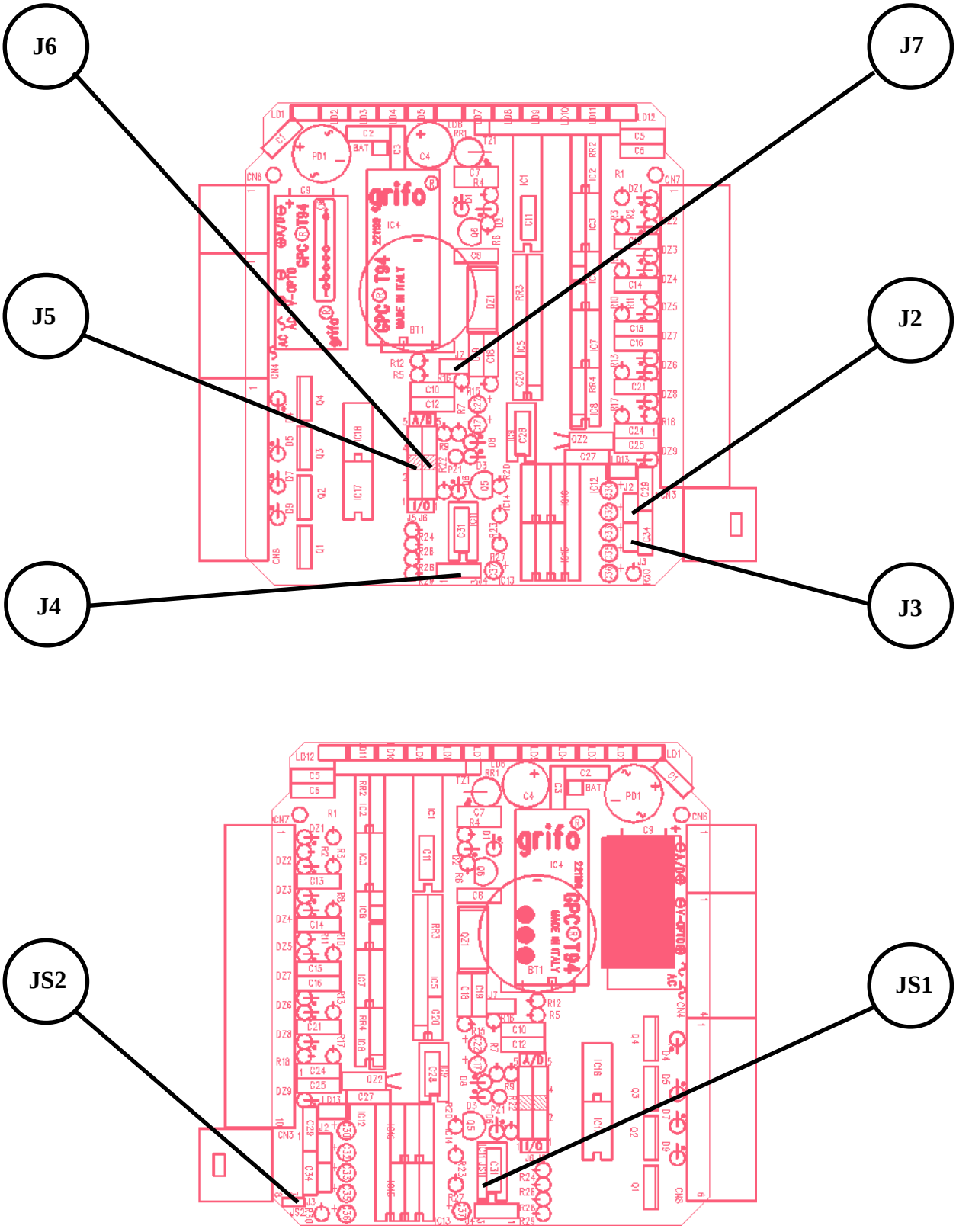


FIGURE 36: GPC® T94 JUMPERS LOCATION

SERIAL COMMUNICATION SELECTION

GPC® R/T 94 serial line can be buffered as TTL, RS 232, RS 422, RS 485 Current Loop. By hardware can be selected which one of these electric standards is used, through jumpers connection and drivers installation. By software the serial line can be programmed to operate with all the standard physical protocols, in fact the bits per character, parity, stop bits and baud rates can be decided by setting opportunes CPU internal registers. In the following paragraphs there are all the informations on serial communication configurations.
Some devices needed for RS 422, RS 485 and Current Loop configurations are not mounted on the board in standard configuration; this is why each fist non-standard (non-RS 232) serial configuration must be always performed by **grifo®** technicians. This far the User can change in autonomy the configuration following the informations below:

- SERIAL LINE IN RS 232 (option **.RS232**)

			IC12	= driver MAX 202
J2, J3	=	not connected	IC13	= no device
J4	=	don't care	IC14	= no device
JS1, JS2	=	not connected	IC15	= no device
			IC16	= no device

- SERIAL LINE B IN CURRENT LOOP (option **.CLOOP**)

			IC12	= no device
J2, J3	=	not connected	IC13	= driver HP 4200
J4	=	don't care	IC14	= driver HP 4100
JS1, JS2	=	not connected	IC15	= no device
			IC16	= no device

Please remark that Current Loop serial interface is passive, so it must be connected an active Current Loop serial line, that is a line provided with its own power supply. Current Loop interface can be employed to make both point-to-point and multi-point connections through a 2-wires or a 4-wires connection.

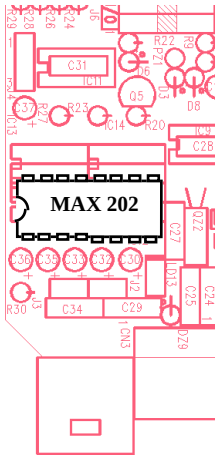
- SERIAL LINE B IN RS 422 (option **.RS 422**)

			IC12	= no device
J2, J3	=	not connected	IC13	= no device
J4	=	position 2-3	IC14	= no device
JS1, JS2	=	(*1)	IC15	= driver MAX 483 or SN 75176
			IC16	= driver MAX 483 or SN 75176

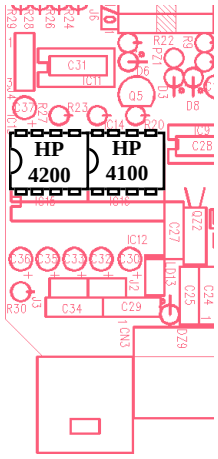
Status of signal P3.7, which is software managed, allows to enable or disable the transmitter as follows:

P3.7= low level	= logic state 0	->	transmitter enabled
P3.7= high level	= logic state 1	->	transmitter disabled

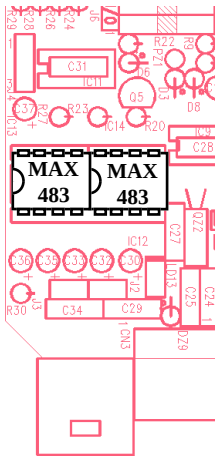
In point-to-point connections, signal P3.7 can be always kept low (trasnmitter always enabled), while in multi-point connections transmitter must be enabled only when a transmission is requested.



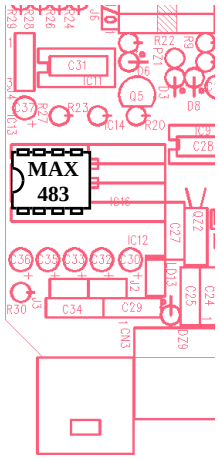
Serial line in RS 232



Serial line in Current Loop



Serial line in RS 422



Serial line in RS 485

FIGURE 37: SERIAL COMMUNICATION DRIVERS LOCATION

- SERIAL LINE B IN RS 485 (option **.RS 485**)

			IC12	=	no device
J2, J3	=	not connected	IC13	=	no device
J4	=	position 1-2	IC14	=	no device
JS1, JS2	=	(*1)	IC15	=	driver MAX 483 or SN 75176
			IC16	=	no device

In this modality the signals to use are pins 5 and 6 of connector CN3, that become transmission or reception lines according to the status of signal P3.7, managed by software, as follows:

P3.7= low level	= logic state 0	->	transmitter enabled
P3.7= high level	= logic state 1	->	transmitter disabled

This kind of serial communication can be used for multi-point connections, in addition it is possible to listen to own transmission, so the User is allowed to verify the succes of transmission. In fact, any conflict on the linecan be recognized by testing the received character after each transmission.

(*1) If using the RS 422 or RS 485 serial line, it is possible to connect the terminating and forcing circuit on the line by using JS1 and JS2. This circuit must be always connected in case of point-to-point connections, while in case of multi-point connections it must be connected olny in the farest boards, that is on the edges of the commmunication line.

- SERIAL LINE IN TTL (default configuration)

			IC12	=	no device
J2, J3	=	connected	IC13	=	no device
J4	=	indifferent	IC14	=	no device
JS1, JS2	=	not connected	IC15	=	no device

When a reset or a power on occour, signal PWM1 is kept to a logic level high, so in one of these two cases driver RS 485 is receiving or RS 422 driver is disabled, avoidng eventual conflicts in communication.
For further informations about serial communication please refer to the examples of figures 12÷18 and to appendix B of this manual.

SOFTWARE

A wide selection of software development tools can be obtained, allowing use of the **GPC® R/T 94** card as a system for its own development, both in assembler and in other high level languages; in this way the user can easily develop all the requested application programs in a very short time. Generally all software packages available for the51 microprocessors family can be used.

MICRO/ASM-51: macro cross assembler available for MS-DOS operating system in "ABSOLUTE" and "RELOCATABLE" version. In this "RELOCATABLE" version is supplied with a linker and a library manager.

MICRO/C-51: integer cross compiler for source files in standard ANSI C, available also for MS-DOS operating system. It produces a source assembly file compatible with MICRO/ASM-51 or with Intel macro relocatable assembler MCS 51.

MICRO/SLD-51: source level debugger and simulator. It is executed on P.C. without any additional hardware and it allows loading of HEX and SYMBOLIC files, breakpoint setting, instruction execution in trace mode, registers and memory dump, etc.

HI-TECH C: cross compiler for C source program. It is a powerful software tool that includes editor, C compiler, assembler, optimizer, linker, library, and remote symbolic debugger, in one easy to use integrated development environment.

BASCOM LT: cross compiler for BASIC source program. It is a powerful software tool that includes editor, BASIC compiler and simulator included in an easy to use integrated development environment for Windows.

DDS MICRO C 51: low cost ross compiler for C source program. It is a powerful software tool that includes editor, C compiler (integer), assembler, optimizer, linker, library, and remote debugger, in one easy to use integrated development environment. There are also included the library sources and many utilities programs.

PERIPHERAL DEVICES SOFTWARE DESCRIPTION

In the previous paragraphs are described the external registers addresses, while in this one there is a specific description of registers meaning and function (please refer to I/O addresses table, for the registers names and addresses values). For a more detailed description of the devices, please refer to manufacturing company documentation; for microprocessor internal peripheral devices, not described in this paragraph, refer to appendix B. In the following paragraphs the **D7÷D0** and **.0÷7** indications denote the eight bits of the combination used in I/O operations.

SERIAL EEPROM

For software management of serial EEPROM module of IC 11, please refer to specific documentation or to demo programs supplied with the card. The first 32 (0÷31) bytes of serial EEPROM are reserved for software tools use, so they can't be neither read nor written by User program. The User must realize a serial communication with I²C bus standard protocol, through two I/O microprocessor pins. The only necessary information is the electric connection:

P1.3	<->	DATA line (**SDA**)
P1.2	->	CLOCK line (**SCL**)

As the hardware implementation of the serial EEPROM module manager circuit, the slave address signals **A0** and **A1** are connected to a logical **0**, while signal **A2** is connected to logical **1**. Logic state 0 corresponds to low level (=0 V), logic state 1 corresponds to high level (=5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the two pins of EEPROM are set to high.

BACKED SRAM + SERIAL RTC

For software management of serial SRAM+RTC module of IC 9, please refer to specific documentation or to demo programs supplied with the card. The User must realize a serial communication with I²C bus standard protocol, through two I/O microprocessor pins. The only necessary information is the electric connection:

P1.3	<->	DATA line (**SDA**)
P1.2	->	CLOCK line (**SCL**)

As the hardware implementation of the serial EEPROM module manager circuit, the slave address signal **A0** is connected to a logical **0**. Logic state 0 corresponds to low level (=0 V), logic state 1 corresponds to high level (=5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the two pins of SRAM+RTC are set to high.

DIGITAL INPUTS

The status of the 9 NPN digital inputs is acquired through the driver serializer 74 HCT 166 (inputs IN0÷7) or directly through a microcontroller I/O pin (input AUX), with the following correspondance:

Line **H** 74 HCT 166	<-	Input IN0
Line **G** 74 HCT 166	<-	Input IN1
Line **F** 74 HCT 166	<-	Input IN2
Line **E** 74 HCT 166	<-	Input IN3
Line **D** 74 HCT 166	<-	Input IN4
Line **C** 74 HCT 166	<-	Input IN5
Line **B** 74 HCT 166	<-	Input IN6
Line **A** 74 HCT 166	<-	Input IN7
Pin **P3.2** (**/INT0**) CPU	<-	Input AUX

When an NPN input is actived (contact closed to the GND opto signal) the corresponding line is at a low logic state (logic 0), vicecersa when an input is not actived (contact open) a high logic level is acquired (logic 1).
As previously stated, LEDs LD6÷12 give the User a visual indication of the digital inputs status (LED on=input actived).
For informations about the management of the 74HCT166 serializer (IC1) please refer to the manifacturer documentation. This technical manual reports no software information about this component because its management requires a deep knowledge of it and anyway the user can take advantage of the special high level procedures delivered with the programming package. Control logic allows the software management of this microprocessor through three I/O microprocessor pins, with the following correspondance:

P3.3	<-	DATA signal
P3.4	->	LOAD signal
P3.5	->	signal CLK

Logic state 0 corresponds to low level (=0 V), logic state 1 corresponds to high level (=5 V). During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the pins of 74 HCT 166 are set to high.

As an example, here is reported a procedure written for **BASCOM 8051** compiler, that returns the status of digital inputs IN0÷7 through 74 HCT 166.

```
' Constants declaration
Dt_166 Alias P3.3          ' 74 HCT 166 DATA signal
Ld_166 Alias P3.4          ' 74 HCT 166 LOAD signal
Clk_166 Alias P3.5         ' 74 HCT 166 CLK signal

' Variables declaration
Dim In0_7 As Byte          ' Status of inputs IN0..IN7
Dim I As Byte              ' General purpose counter
Dim Flg As Bit             ' General purpose flag
```

```
' Procedures declaration
Declare Sub Init_166
Declare Sub Inp_166


' Procedure: INIT_166
' Initializes 74 HCT 166 management lines
' Input:  --
' Output: --

Sub Init_166
  Clk_166 = 1          ' Sets Clock to 1
  Ld_166 = 1           ' Load signal unactived
  Dt_166 = 1           ' Sets Data signal as input
  Flg = Dt_166
Return


" Procedure: INP_166
' REturn the status of the inputs IN0..IN7
' Input:  --
' Output: In0_7 = Status of IN0 (bit 0)..IN7 (bit7)

Sub Inp_166
  In0_7 = 0            ' Resets the variable
  Clk_166 = 0          ' Clock starting status
  Ld_166 = 0           ' Must acquire byte in input
  Clk_166 = 1          ' Clock front to acquire data
  Ld_166 = 1
  For I = 1 To 8       ' Loop to acquire 8 bits
    Flg = Dt_166       ' Read one bit
    Clk_166 = 0        ' Sets Clock to 0
    If Flg = 1 Then    ' Sets bit In0_7.7 to the status of Flg
      Set In0_7.7
    Else
      Reset In0_7.7
    End If
    If I <> 8 Then
      clr c
      Rotate In0_7 , Right , 1      ' Rights shifts the byte of 1 position
      Clk_166 = 1                   ' Clock front for next reading
    End If
  Next I
Return
```

DIGITAL OUTPUTS

The status of the four digital outputs, realys or NPN transistors, is defined through the management of four microcontrolles I/O pins, with the following correspondance:

P1.4	->	Output OUT0	(relay RL1 or transistor Q1)
P1.5	->	Output OUT1	(relay RL2 or transistor Q2)
P1.6	->	Output OUT2	(relay RL3 or transistor Q3)
P1.7	->	Output OUT3	(relay RL4 or transistor Q4)

When the I/O signals are set to logic state low (logic 0), the corresponding output is activated (the transistor is conducting or the relay contact is connected to is common terminal), viceversa when pins are set to logic state high (logic 1), the outputs OUTn are disactivated (the transistor is not conducting or the relay contact is open).
As previoously stated, LEDs LD1÷4 give the User a visual indication of the status of the digital outputs (LED on=output activated).
During a reset or a power on, microprocessor I/O pins are set to logic level 1, so the relays or the transitors are disactived.

JUMPER J1 (RUN/SETUP) - A/D LINE

The status of jumpers J5 and J6 can be acquired by software. In fact when connected it short-circuits the CPU pin **P1.1** (AN1) and pin **P1.0** (AN0). This featues allows the User to implement two system configurations but only if the above mentioned pins are not used to manage the A/D conversion line, the analog comparator or the I/O TTL available on the **GPC® R/T94** board. For example, firmware **ALB x94** uses this feature to distinguish between RUN mode and SETUP mode.
Under the point of view of software management , being the procedure to acquire such configuration input quite articolated, in case of need please contact **grifo®**.

CPU PERIPHERALS

A description of all the microprocessor CPU and registers (Timer Counter, interrupts controller, serial line, I/O ports, etc.) is available in appendix B of this manual. Should such informations be insufficent, please refer to the manifacturer documentation.

A/D CONVERTER

For the A/D Converter section, the conversion is performed through the analog comparator and a timer on board of the microcontroller. For the management of their registers, please refer to appendix B of this manual.
About the software management of the conversion, this technical manual reports no software information about this operation because its management requires a deep knowledge, also theoretical, of it and anyway the User can take advantage of the ATMEL application notes, available on Internet or in the Data Book or use the specific procedures provided with high-level languages software packages.

As an example, here is reorted a procedure written for **BASCOM 8051** compiler, that returns the digital combination corresponding to the input analog signal itself.
In this example timer 0 is used to generate the time base.

```
' Constants declaration
Tr0 Alias Tcon.4          ' Flag start/stop of TIMER 0
Tf0 Alias Tcon.5          ' Flag Over-flow of TIMER 0
An0 Alias P1.0            ' A/D charge capacity input
An1 Alias P1.1            ' A/D analog input

' Procedures declaration
Declare Sub AD_Conv


' Procedure: AD_CONV
' Returns the conversion acquired by the A/D section in 60 msec max
' Input:  --
' Output: TH0, TL0 = Combination acquired

Sub AD_Conv
  Tr0 = 0                 ' Stops Timer 0
  Tf0 = 0                 ' Resets Over-flow flag
  Th0 = 0                 ' Reset Timer 0 registers
  Tl0 = 0
  An1 = 1                 ' Unlocks A/D analog input
  An0 = 1                 ' Starts to charge the capacitor
  Tr0 = 1                 ' Starts Timer 0
                          ' Assembler routine starts
!Ad_2:
  jb tf0 , ad_3  ;        ' Checks conversion end by Over-flow
  jb p3.6 , ad_3 ;        ' Checks conversion end by testing if
                          ' bit P3.6 (comparator output) is = 1
  Sjmp ad_2
!Ad_3:
                          ' Assembler routine ends
  Tr0 = 0                 ' Stopsl Timer 0
```

```
  If Tf0 = 1 Then       ' If Timer 0 is in Over-flow, sets....
    Th0 = 255           ' ....conversion value to full range
    Tl0 = 255
  End If
  Tf0 = 0               ' Resets Over-flow flag
  An0 = 0               ' Discharges the capacitor for about 6 msec
  Waitms 6
End Sub
```

ALB (ABACO® LINK BUS) SOFTWARE DESCRIPTION

The **ALB** communication protocol allows to take advantage of all the on board resources by means of a set of commands; all the characters received on the serial communication line are interpreted and executed, then an eventual answer can beretransmitted to the master control unit. The firmware features also a Setup mode, which allows the user to configure every section of the device.

WORKING MODE SELECTION

The firmware of **GPC® R/T94** board can manage two different working modes, these are **Setup mode** or **Run mode (ALB)**. The selection of which mode to employ happens during the **Power-ON** phase, by testin the status of jumpers **J5** and **J6**:

SETUP mode:	*J5 and J6 in position 2-3 and 4-5* *Pin 3-3 of J5 and J6 CONNECTED*
RUN mode:	*J5 and J6 in any position* *Pin 3-3 of J5 and J6 NOT CONNECTED*

SETUP MODE

In **SETUP** mode it is possible to set the initialization parameters, that is the baud rate, the communication mode and the device name. These settings will be stored in the EEPROM, and will make the working configuration in **RUN** mode. To correctly set the initialization parameters please follow the instructions below:

1) Turn off **GPC® R/T 94**.

2) Be sure that no signal is connected to inputs and outputs.

3) Connect jumpers **J5** and **J6** to select SETUP mode as described in the previous paragraph.

4) Supply the board.

5) If **SETUP** mode has been recognized then the output OUT0 has been activated (LD1 is ON). At this point the User can set baud rate and communication mode configuring IN0÷IN7 inputs as described in the following tables.

NOTE
The desired setting must be kept up to the following point.

BAUD RATE	IN0	IN1	IN2	IN3	IN4	IN5
38400 Baud	OFF	OFF	OFF	OFF	OFF	OFF
19200 Baud	ON	OFF	OFF	OFF	OFF	OFF
9600 Baud	OFF	ON	OFF	OFF	OFF	OFF
4800 Baud	OFF	OFF	ON	OFF	OFF	OFF
2400 Baud	OFF	OFF	OFF	ON	OFF	OFF
1200 Baud	OFF	OFF	OFF	OFF	ON	OFF

FIGURE 38: BAUD RATE SELECTION TABLE

COMMUNICATION TYPE	IN6	IN7
Point to Point	OFF	OFF
Master Slave 9 bits	ON	OFF

FIGURE 39: COMMUNICATION MODES SELECTION TABLE

ON and OFF indicate respectively an input connected and not connected to GND opto signal.
To confirm and store the selected parameters, the User must close the AUX input to GND opto signal and reopen it.

6) If the previous operation didn't succeed then also ouptut OUT3 is activated (LD4 ON) and the User must repeat the operation at point 5. If, otherwise, the operation secceeded, the firmware activates output OUT1 (LD2 ON), to tell that it is possible to set the NAME used by the board for Master Slave serial communication. To set the NAME (permitted values are in the range 128÷255) the same technique described at point 2 should be used, that is, to set an opportune input configuration on signals IN0÷IN7 then to send an impulse to input AUX to confirm the data. Please consider that the NAME is input following a binary codification (input INx in position **ON** means the corresponding bit set to **0**), as reported in the following table, and that the NAME is not significant if the point to point communication mode has been selected.

NAME	IN0	IN1	IN2	IN3	IN4	IN5	IN6	IN7
128	ON	ON	ON	ON	ON	ON	ON	OFF
129	OFF	ON	ON	ON	ON	ON	ON	OFF
...	...	...	...	...	...	...	...	...
255	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

FIGURE 40: NAME SETTING TABLE

7) If the previous operation didn't succeed then also ouptut OUT3 is activated (LD4 ON) and the User must repeat the operation at point 6. If, otherwise, the operation secceeded, the firmware stores all the input parameters into the EEPROM and starts an infinite loop. If this last operation didn't succeed all the OUT0÷OUT3 outputs are activated (LD1÷LD4 ON), otherwise the firmware states the end of the **SETUP** procedure setting the outputs in the following configuration:

OUT0	=	Activated	(LD1 ON)
OUT1	=	Deactivated	(LD2 OFF)
OUT2	=	Activated	(LD3 ON)
OUT3	=	Deactivated	(LD4 OFF)

8) Turn off power supply

9) Connect jumpers J5 and J6 to select **RUN** mode.

RUN MODE

When entering in **RUN** mode the baud rate and communication protocol parameters stored in EEPROM are verified. If they are not valid (e.g. EEPROM not initialized) the board starts a stand-by loop; at this point the User may only turn off the device.

NOTE

EEPROM is NOT initialized by default, the the user must initialize it (SETUP mode) before attempting to use the board.

Baud rate and communication protocol for the **RUN** mode are settable (in **SETUP** mode), while data format is function of the selected communication mde, as follows:

Point to Point Communication:	**8 bit, 1 Stop, NO Parity**
Master Slave:	**9 bit, 1 Stop, NO Parity**

In RUN mode the board's peripherals are managed in the following modalities.

Serial line:
Used to receive commands and send eventual responses; it is possible to communicate in point to point or master slave mode. This last mode allows to make a remote control network where the signle modules are driven by an unique master unit (PC, PLC, **GPC®** card, etc.) also from great distance.

Digital outputs OUT0÷OUT3:
They all can be set at the same time by an unique command, or they can be managed bit level. In this case there are also commands to make timed activations, square waves, etc.

Digital inputs IN0÷IN7:
They are acquired at the same time by an unique command, or they can be managed bit level.

Digital input AUX:
It can be used as digital input line, or as a clock signal for a 16 bits counter, managed through specific commands.

Serial EEPROM:
A set of commands that allow to manage the content of the serial EEPROM as 10 characters long messages is available. So becomes possible to store in non volatile memory informations like text strings, settings, parameters, etc.

SRAM+RTC:
A set of commands that allow to set hours, minutes, seconds, day, month, year and day of week of the Real Time Clock is available. It is also possible to read and write bytes of the SRAM that this device provides.

TTL I/O signals:
They implement two different **1-Wire®** BUS protocol lines. Also, a set of commands to manage the devices connectable to such BUS (temperature sensor, memories, Dallas iButton™, etc.) is available.

GENERAL PURPOSE COMMANDS

Here follows the general purpose commands list.

MASTER RESET

Input Sequence:

Dec Code:	**65**	**97**
Hex Code:	**41**	**61**
Mnemonic:	**A**	**a**

Upon the reception of this command the firmware restores the initial condition that happens to be at the Power-ON; in detail:

OUT0÷OUT3:	They are reset and put into initial condition, then they are set to the logic state 0. Eventual timings are interrupted.
16 bits Counter:	It is initialized to 0.

READ FIRMWARE VERSION

Input Sequence:

Dec Code:	**121**
Hex Code:	**79**
Mnemonic:	**y**

Answer Codes:
The two nibbles of the firmware version are returned:

Version X.Y = Y --> <NibL VAL>
<NibH VAL>

CHECK EEPROM READY FOR WRITING

Input Sequence:

Dec Code:	**66**
Hex Code:	**42**
Mnemonic:	**B**

By this command the user may ask the firmaware if it is ready to manage a new EEPROM message, this command must be sent whenever the User needs to send one of the following messages management commands.

Answer Codes:
The firware returns the following codes:

0	EEPROM not ready to manage a new message
1	EEPROM ready to manage a new message

PRESENCE BYTE OUTPUT

Input Sequence:

Dec Code:	**89**	**dat.L**	**dat.H**
Hex Code:	**41**	**dat.L**	**dat.H**
Mnemonic:	**Y**	**ASCII(dat.L)**	**ASCII(dat.H)**

The presence byte of the board is set to the value of the "dat" parameter, which can range in the values **0÷255**.
This byte is allocated in a reserved EEPROM location and, once it has been set, allows to, for example, verify the correct working of **GPC® R/T 94** or check for conflicts on the serial communication line.
Please remark that "dat" parameter must be sent as two nibbles, in detail must be sent the low nibble first (**dat.L**=bits 0÷3), then the high nibble (**dat.H**=bits 4÷7).

NOTE
This command requires to write into the on board serial EEPROM, so before attempting to execute it the User shold be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

Example:
If the User wishes to store the presence byte "65" then he/she will need to send the sequence:
89 1 4.

PRESENCE BYTE INPUT

Input Sequence:

Dec Code:	**121**
Hex Code:	**79**
Mnemonic:	**y**

GPC® R/T 94 sends to the serial port its presence byte.
This command is useful to verify the correct working of **GPC® R/T 94** or check for conflicts on the serial communication line.

Answer Codes:
A value ranging from 0 to 255 is returned by sending two nibbles in the format seen for the previous command: low nibble first (**dat.L**=bits 0÷3), then the high nibble (**dat.H**=bits 4÷7).

AUX SIGNAL CONFIGURATION SETTING

Input Sequence:

Dec Code:	**85**	**cfg**
Hex Code:	**55**	**cfg**
Mnemonic:	**U**	**ASCII(cfg)**

The AUX signal may be configured as a general purpose INPUT or as a trigger input for the 16 bits internal counter according to the value of **cfg**:

0	*AUX signal is set as an INPUT*
2	*AUX signal is used to Trigger the 16 bit Counter*

If the sequence contains invalid data the command is ignored.

NOTE
This command requires to write into the on board EEPROM, so before attempting to execute it the User shold be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

Example:
If the User wants to set AUX signal as COUNTER then he/she will need to send the following sequence: **85 2.**

AUX SIGNAL CONFIGURATION ACQUISITION

Input Sequence:

Dec Code:	**102**
Hex Code:	**66**
Mnemonic:	**f**

Answer Codes:
The AUX signal configuration byte returned by the firmware may assume one of the following values:

0	*AUX signal is set as an INPUT*
2	*AUX signal is used to Trigger the 16 bit Counter*

READ BOARD CODE

Input Sequence:

Dec Code:	**75**
Hex Code:	**4B**
Mnemonic:	**K**

Answer Codes:
Tha value **1**, which is the **GPC® R/T 94** identificative code, is returned.

DIGITAL I/O PORT BYTE MANAGEMENT COMMANDS

Here follow the commands to manage byte-level the digital I/O ports of **GPC® R/T 94.**

OUTPUT PORT SET

Input Sequence:

Dec Code:	**87**	**1**	**dat**	**0**
Hex Code:	**57**	**1**	**dat**	**0**
Mnemonic:	**W**	**SOH**	**ASCII(dat)**	**NUL**

The <Data> byte must be sent according to the following format:

(MSB) 0 0 0 0 **OUT3** **OUT2** **OUT1** **OUT0** (LSB)

Where **OUTn** stands for the logic state, **0** (output disabled) or **1** (output activated), that the respective output must get.
If the sequence contains invalid data the command is ignored.

Example:
If the User wanta to activate the **OUT0** and **OUT3** outputs then he/she will need to send the following sequence:
87 1 9 0.

INPUT PORT ACQUISITION

Input Sequence:

Dec Code:	**82**	**0**
Hex Code:	**52**	**0**
Mnemonic:	**R**	**NUL**

The serial data read by the input port is returned.

Answer Codes:
The data acquired by the port is returned as two nibbles in the following format:

Nibble L:	(MSB)	0	0	0	0	**IN3**	**IN2**	**IN1**	**IN0**	(LSB)
Nibble H:		0	0	0	0	**IN7**	**IN6**	**IN5**	**IN4**	

Where **INn** stands for the logic state, **0** (ON = input connected to GND opto) or **1** (OFF = input open), that the respective NPN lines have.
If the sequence contains invalid data the command is ignored.

Example:
If the User wants to read the input port, where the **90** (5A Hex) data is present then he/shewill need to send the following sequence: **82 0**,obtaining the following answer: **10 5.**

DIGITAL I/O PORT BIT MANAGEMENT COMMANDS

Here follow the commands to manage bit-level the digital I/O ports of **GPC® R/T 94.**

OUTPUT BIT SET

Input Sequence:

Dec Code:	**83**	**1**	**bit**
Hex Code:	**53**	**1**	**bit**
Mnemonic:	**S**	**SOH**	**ASCII(bit)**

The OUTn output indicated by **bit** gets the logic state **1** (output activated); **bit** can be **0**, **1**, **2** or **3**.
Eventual timings occourring on the output line are interrupted.
If the sequence contains invalid data the command is ignored.

Example:
If you want to activate the **OUT2** output you will need to send the following sequence:
83 1 2.

TIMED OUTPUT BIT SET

Input Sequence:

Dec Code:	**115**	**1**	**bit**	**tmp.L**	**tmp.H**
Hex Code:	**73**	**1**	**bit**	**tmp.L**	**tmp.H**
Mnemonic:	**s**	**SOH**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**

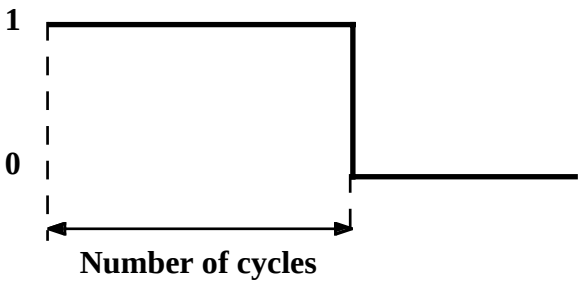


FIGURE 41: TIMED SET COMMAND

The OUTn output indicated by **bit** gets the logic state **1** (line activated); **bit** can be **0**, **1**, **2** or **3**.
The selected output holds the logic state 1 for a number of cycles (1 cycle = **10 msec**) determined by the **tmp** byte, then it returns to the logic state **0** (line disabled). The timing value must range 1÷255 and must be sent as nibbles in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Example:
If the User want to activate the **OUT2** output for 500 msec, corresponding to **50** cycles, then he/she will need to send the following sequence:
115 1 2 2 3.

OUTPUT BIT CLEAR

Input Sequence:

Dec Code:	**67**	**1**	**bit**
Hex Code:	**43**	**1**	**bit**
Mnemonic:	**C**	**SOH**	**ASCII(bit)**

The OUTn output indicated by **bit** gets the logic state **0** (output disabled); **bit** can be **0**, **1**, **2** or **3**.
Eventual timings occourring on the output line are interrupted.
If the sequence contains invalid data the command is ignored.

Example:
If you want to deactivate the **OUT2** output, you will need to send the following sequence:
67 1 2.

TIMED OUTPUT BIT CLEAR

Input Sequence:

Dec Code:	**99**	**1**	**bit**	**tmp.L**	**tmp.H**
Hex Code:	**63**	**1**	**bit**	**tmp.L**	**tmp.H**
Mnemonic:	**c**	**SOH**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**

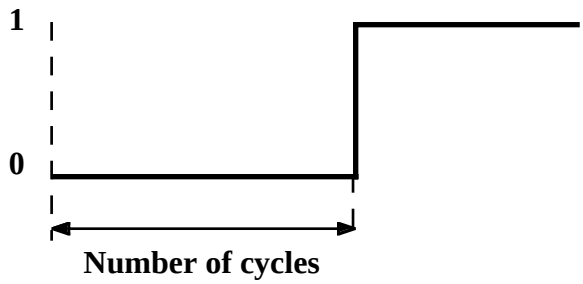


FIGURE 42: TIMED CLEAR COMMAND

The OUTn output indicated by **bit** gets the logic state **0** (line disabled); **bit** can be **0**, **1**, **2** or **3**.
The selected output holds the logic state **0** for a number of cycles (1 cycle = **10 msec**) determined by the **tmp** byte, then it returns to the logic state **1** (line activated). The timing value must range 1÷255 and must be sent as nibbles in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Example:
If the User wants to deactivate the **OUT2** output for 500 msec, that is **50** cycles, then he/she will need to send the following sequence:
99 1 2 2 3.

INPUT BIT OR AUX INPUT ACQUISITION

Input Sequence:

Dec Code:	**114**	**port**	**bit**
Hex Code:	**72**	**port**	**bit**
Mnemonic:	**r**	**ASCII(port)**	**ASCII(bit)**

The logic state of the digital NPN input indicated by the parameters **port** and **bit** is acquired and returned as follows:

Input IN0 ÷ IN7	->	***port** = 0*	***bit** = 0 ÷ 7*
Input AUX	->	***port** = 3*	***bit** = 0*

If the sequence contains invalid data the command is ignored.

Answer codes:
A byte reporting logic state of the digital NPN input indicated is returned; in particular the value **0** (input connected to GND opto) or **1** (input open) can be returned.

Example:
If the User wants to read the **AUX** input, then he/she will need to send: **114 3 0**

DEBOUNCED READ OF AN INPUT PORT BIT

Input Sequence:

Dec Code:	**68**	**port** **bit**	**tmp.L**	**tmp.H**	
Hex Code:	**44**	**port** **bit**	**tmp.L**	**tmp.H**	
Mnemonic:	**D**	**ASCII(port)**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**

The logic state of the digital NPN input indicated by the parameters **port** and **bit** is acquired and returned as follows:

Input IN0 ÷ IN7	->	***port** = 0*	***bit** = 0 ÷ 7*
Input AUX	->	***port** = 3*	***bit** = 0*

Differently from the previous command, the acqusition here is performed through a debouncing process whose duration is specified by parameter **tmp** and expressed in cycles (1 cycle = **10 msec**). The timing value must range 1÷255 and must be sent as nibbles, in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Answer codes:
The data returned may assume one of the following values:

0	*Input closed to GND opto during the whole debouncing*
1	*Input open during the whole debouncing*
7	*Input changed its status during debouncing*

Example:
If the User wants to read the **IN2** input with a debouncing time of 50 msec, corresponding to 5 cycles, then he/she will need to send the following sequence: **68 0 2 5 0**

SQUARE WAVE STARTING WITH "1" OUTPUT BIT

Input Sequence:

Dec Code:	**112**	**1**	**bit**	**tmp.L**	**tmp.H**
			sta.L	**sta.H**	
Hex Code:	**70**	**1**	**bit**	**tmp.L**	**tmp.H**
			sta.L	**sta.H**	
Mnemonic:	**p**	**SOH**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**
			ASCII(sta.L)	**ASCII(sta.H)**	

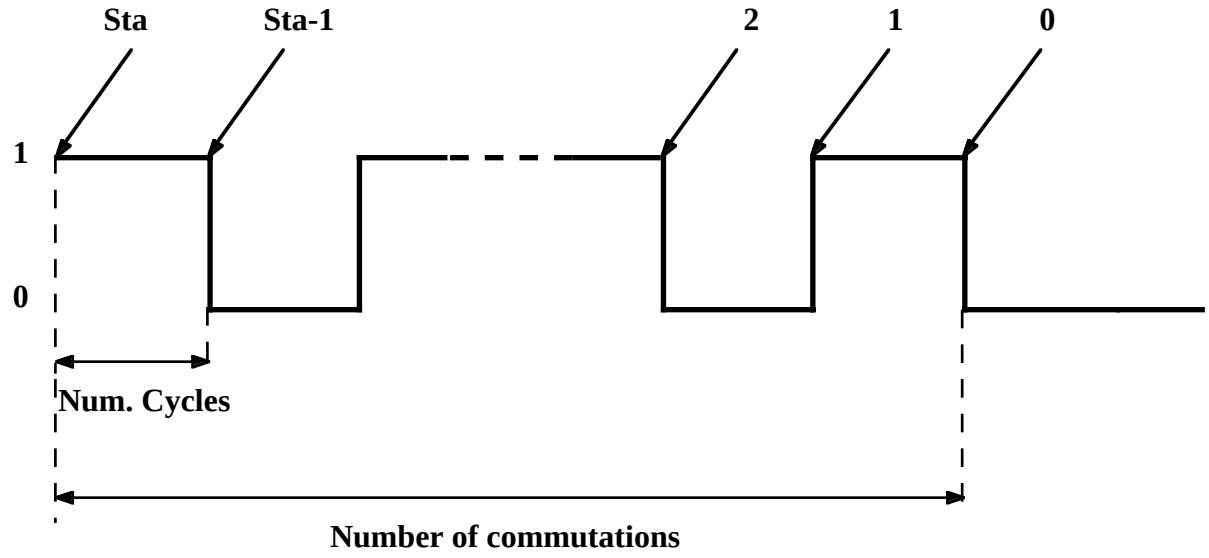


FIGURE 43: TIMED SQUARE WAVE COMMAND

The OUTn output indicated by **bit** outputs a square wave with 50% of Duty Cycle **starting by a logical state "1"**; **bit** can be **0**, **1**, **2** or **3**. The the half-period of the signal contains a number of cycles (1 cycle = **10 msec**) determined by the **tmp** byte. The timing value must range **1÷255** and must be sent as nibbles: in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
The number of commutations that the signal must perform is indicated by the parameter **sta**; as shown by the picture **sta+1** commutations are performed. Also this value must range **1÷255** and must be sent as nibbles: in detail must be sent the low nibble first (**sta.L**=bits 0÷3), then the high nibble (**sta.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Example:
If the User wanta to activate the **OUT2** output for 200 msec, corresponding to **20** cycles and making it commutate **10** times, then he/she will need to send the following sequence:
112 1 2 4 1 9 0.

NOTE
As can be seen by the picture, the final status of the input will depend on the number of commutations performed; in particular an even number of commutations will leave the output in status **0** (output disabled) and viceversa.

SQUARE WAVE STARTING WITH "0" OUTPUT BIT

Input Sequence:

Dec Code:	**119**	**1**	**bit**	**tmp.L**	**tmp.H**
			sta.L	**sta.H**	
Hex Code:	**77**	**1**	**bit**	**tmp.L**	**tmp.H**
			sta.L	**sta.H**	
Mnemonic:	**w**	**SOH**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**
			ASCII(sta.L)	**ASCII(sta.H)**	

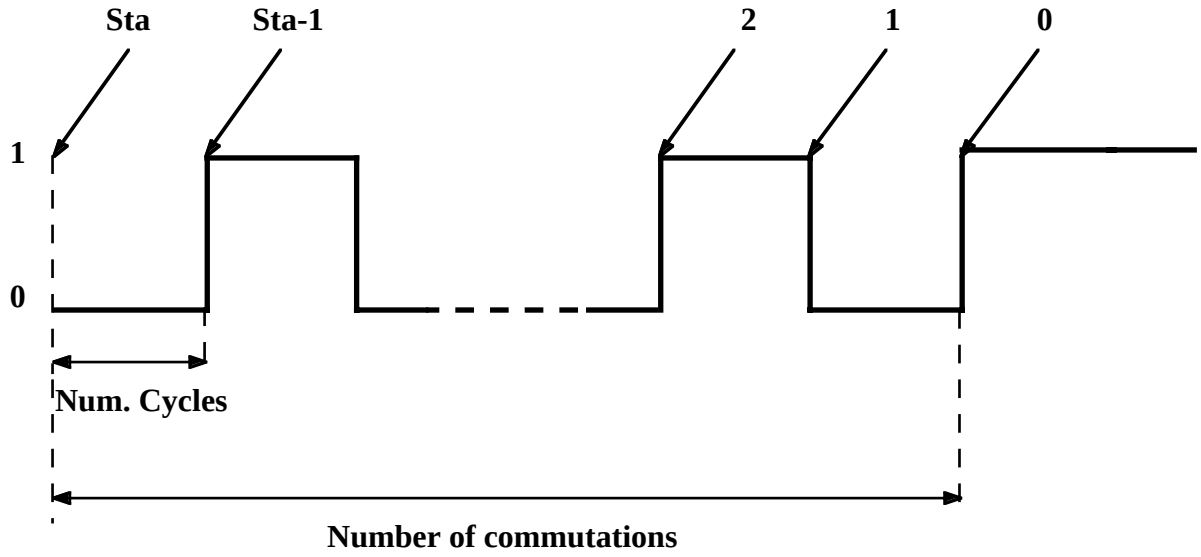


FIGURE 44: TIMED SQUARE WAVE COMMAND

The OUTn output indicated by **bit** outputs a square wave with 50% of Duty Cycle **starting by a logical state "0"**; **bit** can be **0**, **1**, **2** or **3**. The the half-period of the signal contains a number of cycles (1 cycle = **10 msec**) determined by the **tmp** byte. The timing value must range **1÷255** and must be sent as nibbles: in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
The number of commutations that the signal must perform is indicated by the parameter **sta**; as shown by the picture **sta+1** commutations are performed. Also this value must range **1÷255** and must be sent as nibbles: in detail must be sent the low nibble first (**sta.L**=bits 0÷3), then the high nibble (**sta.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

NOTE
As can be seen by the picture, the final status of the input will depend on the number of commutations performed; in particular an even number of commutations will leave the output in status **1** (output disabled) and viceversa.

Example:
If the User wanta to deactivate the **OUT2** output for 200 msec, corresponding to 20 cycles and making it commutate 10 times, then he/she will need to send the following sequence:
119 1 2 4 1 9 0.

SQUARE WAVE OUTPUT BIT

Input Sequence:

Dec Code:	**80**	**1**	**bit**	**tmp.L**	**tmp.H**
Hex Code:	**50**	**1**	**bit**	**tmp.L**	**tmp.H**
Mnemonic:	**P**	**SOH**	**ASCII(bit)**	**ASCII(tmp.L)**	**ASCII(tmp.H)**

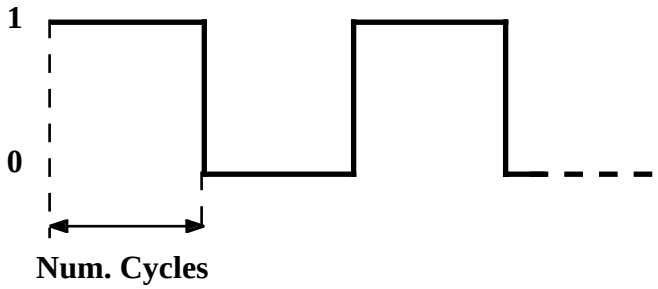


FIGURE 45: SQUARE WAVE COMMAND

The OUTn output indicated by **bit** outputs a square wave with 50% of Duty Cycle; **bit** can be **0**, **1**, **2** or **3**. The the half-period of the signal contains a number of cycles (1 cycle = **10 msec**) determined by the **tmp** byte. The timing value must range **1÷255** and must be sent as nibbles: in detail must be sent the low nibble first (**tmp.L**=bits 0÷3), then the high nibble (**tmp.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Example:
If the User wants to deactivate the **OUT2** output for 200 msec, that is **20** cycles, then he/she will need to send the following sequence: **99 1 2 4 1.**

16 BIT COUNTER MANAGEMENT COMMANDS

Here follow the commands to manage the 16 bit counter. Its value is incremented by the commutations of the AUX signal when this is configured to trigger the counter.

16 BIT COUNTER READ

Input Sequence:

Dec Code:	**73**
Hex Code:	**49**
Mnemonic:	**I**

This command allows to acquire the current value of the 16 bit counter.

Answer Codes:
The sequence returned by the command is made of four bytes showing the 16 bit value currently stored in the counter register; this is sent as nibbles according to the following format:

Counter	(bit 0÷3)	:	(MSB)	0	0	0	0	**Bit3**	**Bit2**	**Bit1**	**Bit0** (LSB)
	(bit 4÷7)	:		0	0	0	0	**Bit7**	**Bit6**	**Bit5**	**Bit4**
	(bit 8÷11)	:		0	0	0	0	**Bit11**	**Bit10**	**Bit9**	**Bit8**
	(bit 12÷15)	:		0	0	0	0	**Bit15**	**Bit14**	**Bit13**	**Bit12**

When the counter reaches the maximum value, which equals to 65535 (FFFF Hex), the next trigger impulse will set the counter to 0. If th eAUX signal is configures as INPUT this command will always return 0.

Example:
If the counter register contains the value 23055 (5A0F Hex), sending the command **73** will return the following values: **15 0 10 5**.

16 BIT COUNTER RESET

Input Sequence:

Dec Code:	**88**	**120**
Hex Code:	**58**	**78**
Mnemonic:	**X**	**x**

Upon the reception of this command the firmware resets the 16 bit counter, which will carry the new value **0**.

MESSAGES MANAGEMENT COMMANDS

LAST MEMORIZABLE MESSAGE ACQUISITION

Input Sequence:

Dec Code: **77**
Hex Code: **4D**
Mnemonic: **M**

This command allows the User to know the maximum number of messages that the board can store. This number depends on the memory device installed according to the following table:

EEPROM	N.MAX
24c02 (256 Bytes)	23
24c04 (512 Bytes)	48
24c08 (1024 Bytes)	99

FIGURE 46: MAXIMUM NUMBER OF MESSAGES MEMORIZABLE IN EEPROM

Answer Codes:
The nuber is returned in two nibbles: in particular first the low nibble will be sent (**carn.L** = bit 0÷3), then the high nibble (**carn.H** = bit 4÷7).

STORING A MESSAGE

Input Sequence:

Dec Code:	**69**	**msg.L**	**msg.H**			
		car0.L	**car0.H**	**...**	**car9.L**	**car9.H**
Hex Code:	**45**	**msg.L**	**msg.H**			
		car0.L	**car0.H**	**...**	**car9.L**	**car9.H**
Mnemonic:	**E**	**ASCII(msg.L)**	**ASCII(msg.H)**			
		ASCII(car0.L)	**ASCII(car0.H)**	**...**	**ASCII(car9.L)**	**ASCII(car9.H)**

The ten characters long message, whose code is indicated by **msg**, is stored in EEPROM. The message number must range **0÷N.MAX** and must be sent in two nibbles: in detail must be sent the low nibble first (**msg.L**=bits 0÷3), then the high nibble (**msg.H**=bits 4÷7).
The value of N.MAX may be acquired by the apposite command.

The characters codes must be in the range **0÷255** (**0÷FF** Hex) and must be sent in nibbles: in detail must be sent the low nibble first (**carn.L**=bits 0÷3), then the high nibble (**carn.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

NOTE
This command requires to write into the on board serial EEPROM, so before attempting to execute it the User shold be sure that such device is ready to be written in, if this condition is not satisfied the command is ignored.

Example:
If the User wants to store the message "ABCDEFGHIJ" (corresponding to the codes: 65, 66, 67, 68, 69, 70, 71, 72, 73, 74) with number 16, then he/she will need to send the following sequence:
69 0 1 1 4 2 4 3 4 4 4 5 4 6 4 7 4 8 4 9 4 10 4.

READING A MESSAGE

Input Sequence:

Dec Code:	**76**	**msg.L**	**msg.H**
Hex Code:	**4C**	**msg.L**	**msg.H**
Mnemonic:	**L**	**ASCII(msg.L)**	**ASCII(msg.H)**

The ten characters long message, whose code is indicated by **msg**, is read from the EEPROM and sent on the serial connection. The message number must range **0÷N.MAX** and must be sent in two nibbles: in detail must be sent the low nibble first (**msg.L**=bits 0÷3), then the high nibble (**msg.H**=bits 4÷7).
The value of N.MAX may be acquired by the apposite command.
If the sequence contains invalid data the command is ignored.

Answer Codes:
The ten characters are returned in nibbles: in detail must be sent the low nibble first (**carn.L**=bits 0÷3), then the high nibble (**carn.H**=bits 4÷7).

If the User wants to read the message with number 16 stored in the previous example, then he/she needs to send the following sequence: **76 0** **1.** The answer will be the sequence:
1 4 2 4 3 4 4 4 5 4 6 4 7 4 8 4 9 4 10 4.

SRAM+RTC MANAGEMENT COMMANDS

CLOCK SETTING

Input Sequence:

Dec Code:	**79**	**ore.L**	**ore.H**	**min.L**	**min.H**
		sec.L	**sec.H**	**gio.L**	**gio.H**
		mes.L	**mes.H**	**ann.L**	**ann.H**
		gse.L	**gse.H**		
Hex Code:	**4F**	**ore.L**	**ore.H**	**min.L**	**min.H**
		sec.L	**sec.H**	**gio.L**	**gio.H**
		mes.L	**mes.H**	**ann.L**	**ann.H**
		gse.L	**gse.H**		
Mnemonic:	**O**	**ASCII(ore.L)**	**ASCII(ore.H)**	**ASCII(min.L)**	**ASCII(min.H)**
		ASCII(sec.L)	**ASCII(sec.H)**	**ASCII(gio.L)**	**ASCII(gio.H)**
		ASCII(mes.L)	**ASCII(mes.H)**	**ASCII(ann.L)**	**ASCII(ann.H)**
		ASCII(gse.L)	**ASCII(gse.H)**		

The Real Time Clock registers are initialized as follows: hours with **ore**, minutes with **min**, seconds with **sec**, day with **gio**, month with **mes**, year with **ann** and day of week with **gse**.
Each of these parameters must be sent in two nibbles: in detail must be sent the low nibble first (**xxx.L**=bits 0÷3), then the high nibble (**xxx.H**=bits 4÷7). The value must be in the respective range as shown in the following table:

BYTE	RANGE	MEANING
ORE	0 ... 23	HOURS
MIN	0 ... 59	MINUTES
SEC	0 ... 59	SECONDS
GIO	1 ... 31	DAY
MES	1... 12	MONTH
ANN	0 ... 99	YEAR
GSE	0 ... 6	Day of week: 0 -> SUNDAY ..... 6 -> SATURDAY

FIGURE 47: RTC INITIALIZATION BYTES VALIDITY RANGE

If the sequence contains invalid data the command is ignored.

Example:
If the User wishes to set the RTC as: **Monday May 11th 1998 12:30:40** the nhe/she will need to send:
79 12 0 14 1 8 2 11 0 5 0 2 6 1 0.

CLOCK READ

Input Sequence:

Dec Code:	**111**
Hex Code:	**6F**
Mnemonic:	**o**

The Real Time Clock registers are read and their content is sent to the serial port.

Answer Codes:
Seven bytes representing hours, minutes, seconds, day, month, year and day of week are returned, following the format shown in the previous table. Each of these bytes is sent in two nibbles: in detail the low nibble is sent first (**xxx.L**=bits 0÷3), then the high nibble is sent (**xxx.H**=bits 4÷7).

RTC SRAM WRITE

Input Sequence:

Dec Code:	**71**	**ind.L**	**ind.H**	**dat.L**	**dat.H**
Hex Code:	**47**	**ind.L**	**ind.H**	**dat.L**	**dat.H**
Mnemonic:	**G**	**ASCII(ind.L)**	**ASCII(ind.H)**	**ASCII(dat.L)**	**ASCII(dat.H)**

The data **dat** (ranging in **0÷255**) is stored in the RTC serial SRAMat the address **ind** (ranging in **32÷255**). Each of these parameters must be sent in two nibbles: in detail must be sent the low nibble first (**xxx.L**=bits 0÷3), then the high nibble (**xxx.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Example:
If the User wishes to store data "65" at the address "100" then he/she will need to send the sequence:
71 4 6 1 4.

RTC SRAM READ

Input Sequence:

Codice Dec:	**103**	**ind.L**	**ind.H**
Codice Hex:	**67**	**ind.L**	**ind.H**
Mnemonico:	**g**	**ASCII(ind.L)**	**ASCII(ind.H)**

The data stored in the RTC serial SRAM at the address **ind** (ranging 32÷255) is read and sent to the serial port. The parameter **ind** is the address to read from. It must be sent in two nibbles: in detail must be sent the low nibble first (**ind.L**=bits 0÷3), then the high nibble (**ind.H**=bits 4÷7).
If the sequence contains invalid data the command is ignored.

Codici di Risposta:
The data byte, ranging in 0÷255, is returned. It is sent in two nibbles: in detail the low nibble is sent first (**dat.L**=bits 0÷3), then the high nibble is sent (**dat.H**=bits 4÷7).

1-WIRE® COMMUNICATION BUS LINES

The TTL I/O lines available on **GPC® R/T 94** are used to implement a **1-Wire®** BUS communication protocol. By means of the commands explained in the following paragraph it is possible to manage the several devices developed to work with this standard (temperature sensors, memories, Dallas iButton™, etc.).
The TTL I/O signals available on connector CN6 are used as follows:

Pin 1 CN6	->	*Line BUS 1-Wire® n.* ***0***
Pin 2 CN6	->	*Line BUS 1-Wire® n.* ***1***

As can be seen in the following paragraph, the high level commands available do not support the presence of more than one device on each BUS, in fact, for example, there is not the command "Search ROM", whose task is to search for ROM codes on the line.
It is possible, anyway, to manage a network connection of many 1-Wire® units taking advantage of the low level commands (BUS reset, get and set a bit status, read and write bytes); in this case the implementation happens to be very articualte and requires also a heavy communication between the master unit and **GPC® R/T 94**.
It is suggestable to connect the board to at most two 1-Wire® devices; the following figure reports a connection example with two temperature sensors Dallas DS18s20.

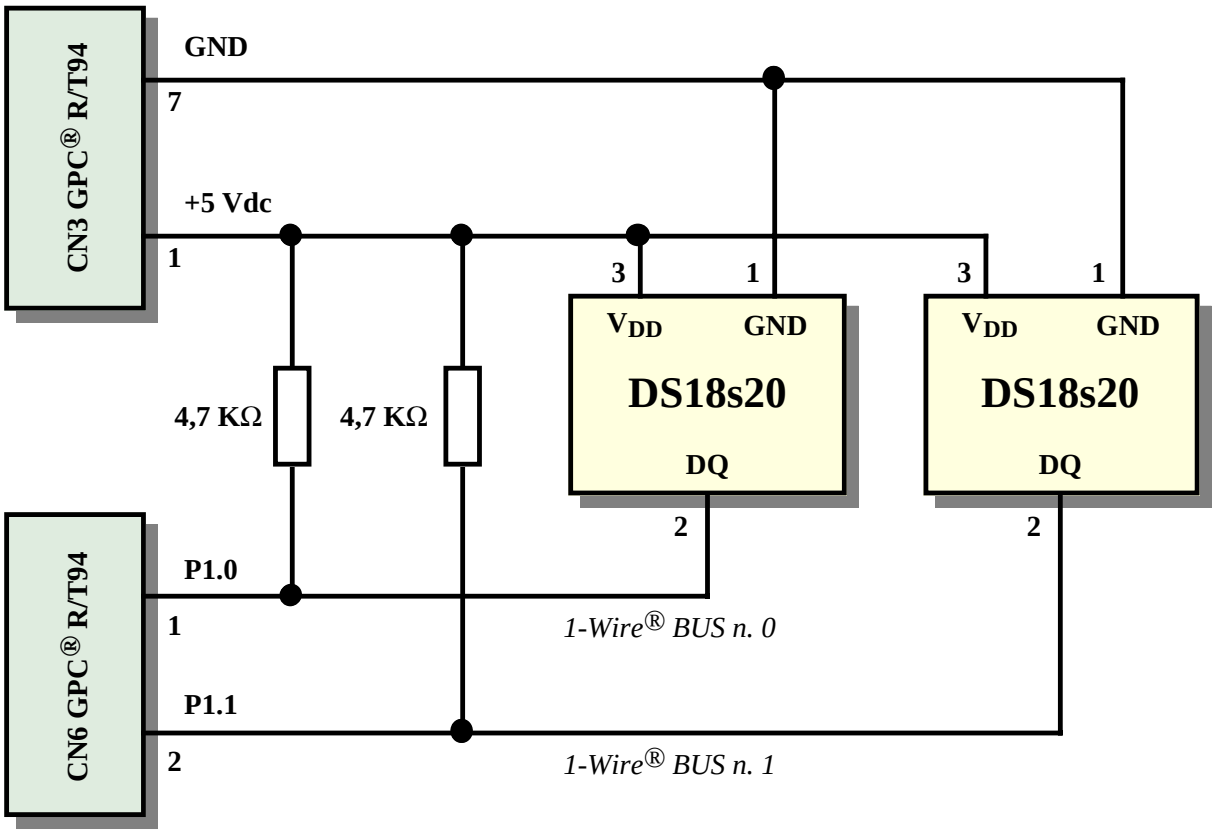


FIGURE 48: CONNECTION EXAMPLE WITH TWO 1-WIRE® DEVICES

NOTE
For a correct use of the two **1-Wire®** lines, jumpers J5 and J6 of **GPC® R/T 94** must be in position **1-2 and 3-4**, to configure the pins of CN6 as generic I/O lines.

COMMANDS TO MANAGE THE 1-WIRE® BUS LINES

Here follow the commands used to manage the two 1-Wire® BUS lines available on **GPC® R/T 94**.

RESET 1-WIRE® BUS AND GET PRESENCE PULSE

Input Sequence:

Dec Code:	**33**	**85**	**wire**
Hex Code:	**21**	**55**	**wire**
Mnemonic:	**!**	**U**	**ASCII(wire)**

The reset sequence is sent to the 1-Wire® BUS line indicated by parameter **wire**.
The value of **wire** must be **0** or **1**, otherwise the command is ignored.
After the reset sequence has finished, the presence impulse of an eventual other device on the line is acquired.

Answer Codes:
The presence impulse logic status is returned, in detail:

0	*1-Wire® device present and ready to receive commands*
1	*1-Wire® device not present*

Example:
If the User wants to send the reset sequence and acquire the presence pulse on 1-Wire® BUS n. 1, then he/she will need to send the following sequence: **33 85 1**

WRITE A BIT TO 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**119**	**wire**	**bit**
Hex Code:	**21**	**77**	**wire**	**bit**
Mnemonic:	**!**	**w**	**ASCII(wire)**	**ASCII(bit)**

The **bit** value specified by parameter bit is sent to the 1-Wire® BUS line indicated by parameter **wire**.
The values of **wire** and **bit** must be **0** or **1**, otherwise the command is ignored.

Example:
If the User wants to send the bit value **1** on 1-Wire® BUS n. 0, then he/she will need to send the following sequence: **33 119 0 1**

READ A BIT FROM 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**114**	**wire**
Hex Code:	**21**	**72**	**wire**
Mnemonic:	**!**	**r**	**ASCII(wire)**

The **bit** value of the 1-Wire® BUS line indicated by parameter **wire** is acquired.
The value of **wire** must be **0** or **1**, otherwise the command is ignored.

Answer Codes:
The logic status (**0** or **1**) of the bit acquired from the specified 1-Wire® BUS line is returned.

Example:
If the User wants to read the status of 1-Wire® BUS n. 0, then he/she will need to send the following sequence: **33 114 0**

WRITE N BYTES TO 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**87**	**wire**		**n**	
		dat0.L	**dat0.H**	**...**	**datn.L**	**datn.H**
Hex Code:	**21**	**57**	**wire**		**n**	
		dat0.L	**dat0.H**	**...**	**datn.L**	**datn.H**
Mnemonic:	**!**	**W**	**ASCII(wire)**		**ASCII(n)**	
		ASCII(dat0.L)	**ASCII(dat0.H)**	**...**	**ASCII(datn.L)**	**ASCII(datn.H)**

The "n" bytes specified by parameter **n** are sent to the 1-Wire® BUS line indicated by parameter **wire**.
The value of **wire** must be **0** or **1**; the maximum number of bytes to send after parameter **n** is **10**, so its value must be in the range **1÷10**.
The "n" bytesvalue must range **0÷255** and must be sent as nibbles, in detail must be sent the low nibble first (**datn.L**=bits 0÷3), then the high nibble (**datn.H**=bits 4÷7).
Command is ignored if the sequence contains invalid data.

Example:
If the User wants to send the bytes 23, 118, 80 and 12 to 1-Wire® BUS n. 1, then he/she will need to send the following sequence: **33 87 1 4 7 1 6 7 0 5 12 0**

WRITE N BYTES TO 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**87**	**wire**	**n**	
Hex Code:	**21**	**57**	**wire**	**n**	
Mnemonic:	**!**	**R**	**ASCII(wire)**		**ASCII(n)**

The "n" bytes specified by **n** are read from the 1-Wire® BUS line indicated by parameter **wire**.

The value of **wire** must be **0** or **1**; the maximum number of bytes readable with one command is **10**, so value of parameter **n** must be in the range **1÷10**.
Command is ignored if the sequence contains invalid data.

Answer Codes:
The n bytes read from the 1-Wire® BUS line indicated are returned as nibbles, in detail will be received the low nibble first (**datx.L**=bits 0÷3), then the high nibble (**datx.H**=bits 4÷7).

Example:
If the User wants to read 4 bytes (that will be 23, 118, 80 and 12) from 1-Wire® BUS number 0, then he/she will need to send the following sequence: **33 8 0 4.**
The response will be: **7 1 6 7 0 5 12 0**

READ ROM CODE FROM 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**76**	**wire**
Hex Code:	**21**	**4C**	**wire**
Mnemonic:	**!**	**L**	**ASCII(wire)**

This command will perform the following operations on the 1-Wire® BUS line specified by parameter **wire**.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receice next command.

b) If the device is present, the **Read ROM** command (code **33 Hex** of 1-Wire® BUS protocol) is sent, then the device ROM code is acquired.

The value of **wire** must be **0** or **1**, otherwise the command is ignored.

Answer Codes:
The response to this command is made of 8 bytes (**rom0÷rom7**) sent as nibbles, in detail will be received the low nibble first (**romx.L**=bits 0÷3), then the high nibble (**romx.H**=bits 4÷7).
Here is the meaning of the bytes:

Device 1-Wire® present and command executed succesfully:
The 8 bytes are the device ROM code: **rom0**=Family code, **rom1÷rom6**=Serial number and **rom7**=CRC.

Device 1-Wire® not present and command not sent:
The 8 bytes are all **0's**.

Example:
If the User wants to acquire the ROM code of the device connected to 1-Wire® BUS number 1, then he/she will need to send the following sequence: **33 76 0.** If the ROM code is: 16 (family code), 56, 198, 13, 0, 8, 0, 226 (CRC), the response to the command wil be:
0 1 8 3 6 12 13 0 0 0 8 0 0 0 2 14

MATCH ROM COMMAND ON 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**77**	**wire**			
		rom0.L	**rom0.H**	**...**	**rom7.L**	**rom7.H**
Codice Hex:	**21**	**4D**	**wire**			
		rom0.L	**rom0.H**	**...**	**rom7.L**	**rom7.H**
Mnemonico:	**!**	**M**	**ASCII(wire)**			
		ASCII(rom0.L)	**ASCII(rom0.H)**	**...**	**ASCII(rom7.L)**	**ASCII(rom7.H)**

This command will perform the following operations on the 1-Wire® BUS line specified by parameter **wire**.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receice next command.

b) If the device is present, the **Match ROM** command (code **55 Hex** of 1-Wire® BUS protocol) is sent, then the device ROM code follows: **rom0**=Family code, **rom1÷rom6**=Serial number, **rom7**=CRC.

The value of **wire** must be **0** or **1**, while the 8 bytes "rom0"÷"rom7" may vary in the range **0÷255** and must be sent as nibbles, in detail must be sent the low nibble first (**romx.L**=bits 0÷3), then the high nibble (**romx.H**=bits 4÷7).
The command is ignored if the sequence contains invalid data.

Answer Codes:
The response to this command can be:

0	*Device 1-Wire® present and command sent*
1	*Device 1-Wire® not present and command not sent*

Example:
If the User wants to send the Match ROM command to 1-Wire® BUS number 1 to match a ROM code: 16 (family code), 56, 198, 13, 0, 8, 0, 226 (CRC), then he/she will have to send the following sequence: **33 77 1 0 1 8 3 6 12 13 0 0 0 8 0 0 0 2 14**

SKIP ROM COMMAND ON 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**83**	**wire**
Codice Hex:	**21**	**53**	**wire**
Mnemonico:	**!**	**S**	**ASCII(wire)**

This command will perform the following operations on the 1-Wire® BUS line specified by parameter **wire**.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receice next command.

b) If the device is present, the **Skip ROM** command (code **CC Hex** of 1-Wire® BUS protocol) is sent.

The value of **wire** must be **0** or **1**, or the command is ignored.

Answer Codes:
The response to this command can be:

0	*Device 1-Wire® present and command sent*
1	*Device 1-Wire® not present and command not sent*

Example:
If the User wants to send the Skip ROM command to 1-Wire® BUS number 1, then he/she will have to send the following sequence: **33 83 1**

ALARM SERCH COMMAND ON 1-WIRE® BUS

Input Sequence:

Dec Code:	**33**	**65**	**wire**
Codice Hex:	**21**	**41**	**wire**
Mnemonico:	**!**	**A**	**ASCII(wire)**

This command will perform the following operations on the 1-Wire® BUS line specified by parameter **wire**.

a) The reset sequence is sent to test the presence of the device on the specified line and to prepare it to receive next command.

b) If the device is present, the **Alarm serch** command (code **EC Hex** of 1-Wire® BUS protocol) is sent.

The value of **wire** must be **0** or **1**, or the command is ignored.

Answer Codes:
The response to this command can be:

0	*Device 1-Wire® present with alarm flag not set*
1	*Device 1-Wire® present with alarm flag set*
7	*Device 1-Wire® not present and command not sent*

Example:
If the User wants to send the Alarm serch command to 1-Wire® BUS number 0, then he/she will have to send the following sequence: **33 65 0**

9 BITS MASTER-SLAVE COMMUNICATION MODE

The Master-Slave communication takes advantage of the 9 bits mode
This means that a nineth bit is used to distinguish between a call from a "**Master**" device to one of the "**Slave**" structures and a mere communication of datas between the Master and the selected Slave. When the nineth bit is set to 1, the data byte must contain the name, or ID code, of the new target device, while if the nineth bit is set to 0 it is possible to send or receive informations from the selected target device.

If the communication is running under **ALB** protocol the ID code must be the byte set in **SETUP mode (NAME).** When this byte is received by a device (**with the nineth bit set to 1**) it recognizes itself and starts to wait for a string containing dats or commands (**with the nineth bit set to 0**); the string must be maximun **24 bytes** long.
It can contain only one command which requires to return an answer code through the serial line, more commands of this kind will be ignored.
The delay between two consecutive characters must be lower than **Time-Out,** because otherwise the string is considered terminated and the answering phase starts.
Here follows the list of Time-Outs related to the Baud Rate:

Baud Rate	**Time-Out**
38400 Baud	*550 µsec*
19200 Baud	*990 µsec*
9600 Baud	*1.54 msec*
4800 Baud	*3.08 msec*
2400 Baud	*6.105 msec*
1200 Baud	*12.1 msec*

When the Time-Out happens, the answer sequence begins; this is made of a byte containing the presence code **6** (**6** Hex), or a data sequence requested by a read command sent during the previous call.

Example:
If a string containing the Port read command is sent, the answer to that call will be the presence code, while the answer to the next call will be the data acquired by the Port sent in the previous request.

After having sent the last character of the string the user will have to wait for a time:

"One char transmission time" + Time-Out

before receiving the first char of the answer sequence.

Example:
At 38.4 KBaud, After having sent the last character of the string the User will have to wait for about 810 µsec before to receive the first char of the answer sequence.

NOTE:

1) Between two calls it is essential to wait for a time that depends on the number of commands sent and the type of operations that they involve to be sure that the transmitted command will be executed correctly.

2) If the master unit is not capable to transfer 9 bits data, it is possible to simumate this kind of communication through the parity bit and programming opportunaly the parity even or odd before transmitting each single byte.

The Byte to transmit has an EVEN number of bits 1

*IF Bit 9 is required to be **1*** -> *Set parity to ODD*
*IF Bit 9 is required to be **0*** -> *Set parity to EVEN*

The Byte to transmit has an ODD number of bits 1

*IF Bit 9 is required to be **1*** -> *Set parity to EVEN*
*IF Bit 9 is required to be **0*** -> *Set parity to ODD*

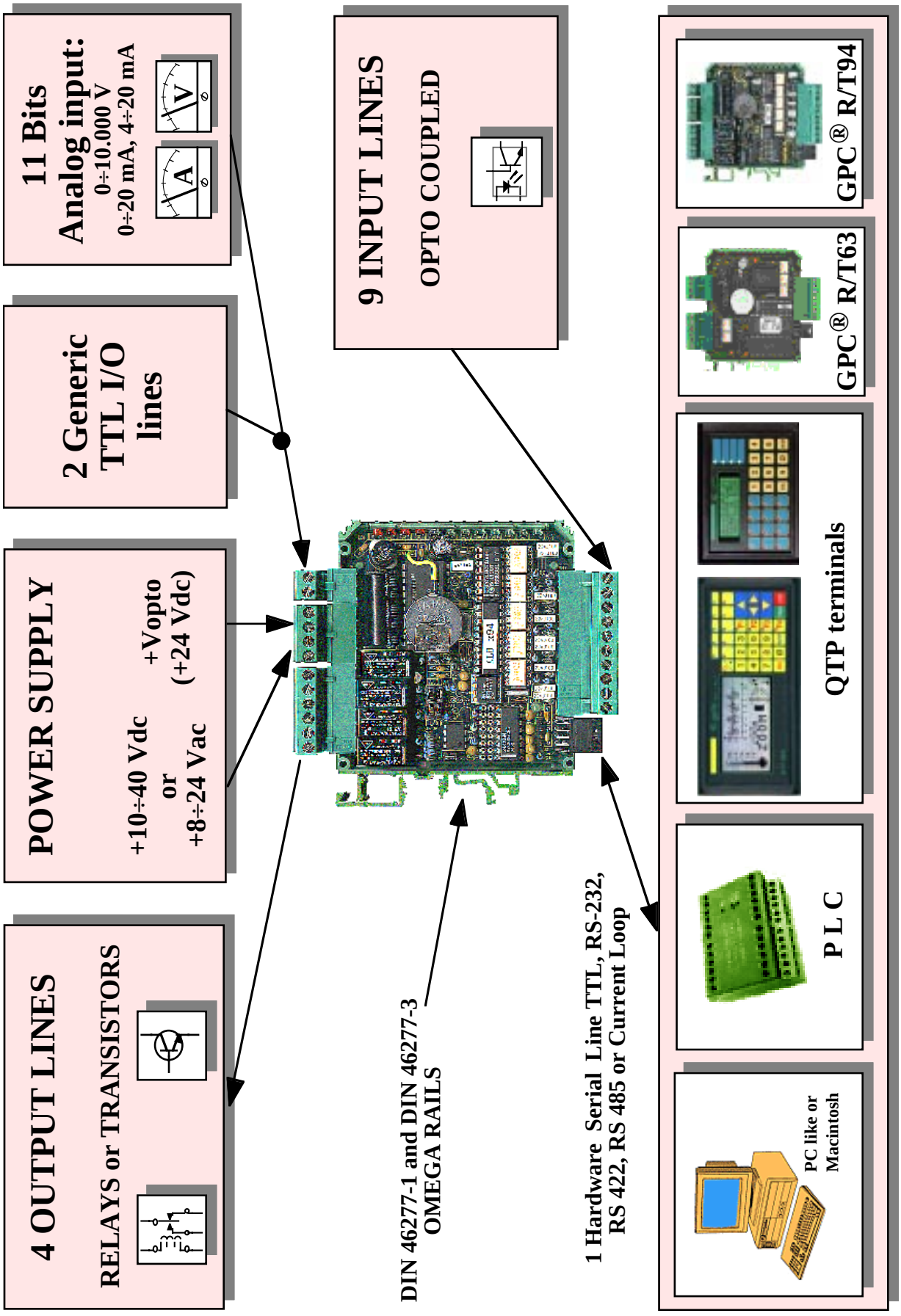


FIGURE 49: POSSIBLE CONNECTIONS DIAGRAM

BIBLIOGRAPHY

In this chapter there is a complete list of technical books, where the user can find all the necessary documentations on the components mounted on **GPC® R/T94**.

Manual TEXAS INSTRUMENTS:	*The TTL Data Book - SN54/74 Families*
Manual TEXAS INSTRUMENTS:	*RS-422 and RS-485 Interface Circuits*
Manual MAXIM:	*New Releases Data Book - Volume 4*
Manual XICOR:	*Data Book*
Manual PHILIPS:	*IC12 - I^2C bus*
Manual ATMEL:	*Microcontroller - AT89 series*
Manual TOSHIBA:	*Photo Couplers - Data Book*
Manual MOTOROLA:	*Bipolar Power Transistor Data*
Manuale HEWLETT PACKARD:	*Optoelectronics Designer's Catalog*
Techincal notes:	*LM2825N*
DALLAS technical notes:	*Documentation and application notes about 1-Wire® communication protocol and devices*

For further information and upgrades please refer to specific internet web pages of the manufacturing companies.

APPENDIX A: COMMANDS OF FIRMWARE ALB X94

Here follow the summarizing tables of firmware **ALB x94** commands.

COMMAND	DEC. Code	HEX Code	MNEMONIC
Master reset	65 97	41 61	A a
Read firmware version	86	56	V
Check EEPROM ready for writing	66	42	B
Presence byte output	89 dat.L dat.H	59 dat.L dat.H	Y ASCII(dat.L) ASCII(dat.H)
Presence byte input	121	79	y
AUX signal configuration setting	85 cfg	55 cfg	U ASCII(cfg)
AUX signal configuration acquisition	102	66	f
Output port set	87 1 dat 0	57 1 dat 0	W SOH ASCII(dat) NUL
Input port acquisition	82 0	52 0	R NUL
Output bit set	83 1 bit	53 1 bit	S SOH ASCII(bit)
Timed output bit set	115 1 bit tmp.L tmp.H	73 1 bit tmp.L tmp.H	s SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)
Output bit clear	67 1 bit	43 1 bit	C SOH ASCII(bit)
Timed output bit clear	99 1 bit tmp.L tmp.H	63 1 bit tmp.L tmp.H	c SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)
Input bit or AUX input acquisition	114 port bit	72 port bit	r ASCII(port) ASCII(bit)
Square wave output bit	80 1 bit tmp.L tmp.H	50 1 bit tmp.L tmp.H	P SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H)
Square wave starting with "1"output bit	112 1 bit tmp.L tmp.H sta.L sta.H	70 1 bit tmp.L tmp.H sta.L sta.H	p SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H) ASCII(sta.L) ASCII(sta.H)
Square wave starting with "0"output bit	119 1 bit tmp.L tmp.H sta.L sta.H	77 1 bit tmp.L tmp.H sta.L sta.H	w SOH ASCII(bit) ASCII(tmp.L) ASCII(tmp.H) ASCII(sta.L) ASCII(sta.H)
16 bit counter read	73	49	I
16 bit counter reset	88 120	58 78	X x

FIGURE A1: ALB X94 COMMANDS SUMMARIZING TABLE 1

COMMAND	DEC. Code	HEX Code	MNEMONIC
Last memorizable message acqisition	77	4D	M
Storing a message	69 msg.L msg.H car0.L car0.H car9.L car9.H	45 msg.L msg.H car0.L car0.H car9.L car9.H	E ASCII(msg.L) ASCII(msg.H) ASCII(car0.L) ASCII(car0.H) ..... ASCII(car9.L) ASCII(car9.H)
Reading a message	msg.L msg.H	4C msg.L msg.H	L ASCII(msg.L) ASCII(msg.H)
Clock setting	79 ore.L ore.H min.L min.H sec.L sec.H gio.L gio.H mes.L mes.H ann.L ann.H gse.L gse.H	4F ore.L ore.H min.L min.H sec.L sec.H gio.L gio.H mes.L mes.H ann.L ann.H gse.L gse.H	O ASCII(ore.L) ASCII(ore.H) ASCII(min.L) ASCII(min.H) ASCII(sec.L) ASCII(sec.H) ASCII(gio.L) ASCII(gio.H) ASCII(mes.L) ASCII(mes.H) ASCII(ann.L) ASCII(ann.H) ASCII(gse.L) ASCII(gse.H)
Clock read	111	6F	o
RTC SRAM write	71 ind.L ind.H dat.L dat.H	47 ind.L ind.H dat.L dat.H	G ASCII(ind.L) ASCII(ind.H) ASCII(dat.L) ASCII(dat.H)
RTC SRAM read	103 ind.L ind.H	67 ind.L ind.H	g ASCII(ind.L) ASCII(ind.H)
Reset BUS 1-Wire®	33 85 wire	21 55 wire	! U ASCII(wire)
WR bit BUS 1-Wire®	33 119 wire bit	21 77 wire bit	! w ASCII(wire) ASCII(bit)
RD bit BUS 1-Wire®	33 114 wire	33 72 wire	! r ASCII(wire)
Write n bytes to BUS 1-Wire®	33 87 wire n dat0.L dat0.H ..... datn.L datn.H	21 57 wire n dat0.L dat0.H ..... datn.L datn.H	! W ASCII(wire) ASCII(n) ASCII(dat0.L) dat(dat0.H) ..... ASCII(datn.L) ASCII(datn.H)
Read n bytes from BUS 1-Wire®	33 82 wire n	21 52 wire n	! R ASCII(wire) ASCII(n)
Read ROM (1-Wire®)	33 76 wire	33 4C wire	! L ASCII(wire)
Match ROM (1-Wire®)	33 77 wire rom0.L rom0.H rom7.L rom7.H	33 4D wire rom0.L rom0.H rom7.L rom7.H	! M ASCII(wire) ASCII(rom0.L) ASCII(rom0.H) ..... ASCII(rom7.L) ASCII(rom7.H)
Skip ROM (1-Wire®)	33 83 wire	33 53 wire	! S ASCII(wire)
Alarm search (1-Wire®)	33 65 wire	33 41 wire	! A ASCII(wire)

FIGURE A2: ALB X94 COMMANDS SUMMARIZING TABLE 2

APPENDIX B: ON BOARD COMPONENTS DESCRIPTION

Features

- **Compatible with MCS-51™ Products**
- **4K Bytes of Reprogrammable Flash Memory**
  - **Endurance: 1,000 Write/Erase Cycles**
- **3.0V to 6V Operating Range**
- **Fully Static Operation: 0 Hz to 24 MHz**
- **Two-Level Program Memory Lock**
- **128 x 8-Bit Internal RAM**
- **15 Programmable I/O Lines**
- **Two 16-Bit Timer/Counters**
- **Six Interrupt Sources**
- **Programmable Serial UART Channel**
- **Direct LED Drive Outputs**
- **On-Chip Analog Comparator**
- **Low Power Idle and Power Down Modes**
- **Brown-Out Detection**

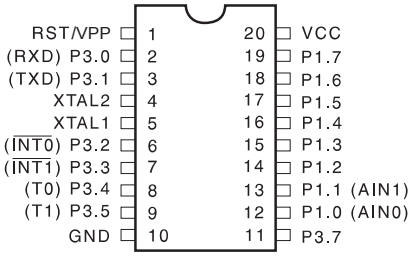
Description

The AT89C4051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 4K Bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C4051 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

The AT89C4051 provides the following standard features: 4K Bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C4051 is designed with static logic for operation down to zero frequency and supports two software-selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

Pin Configuration

PDIP/SOIC

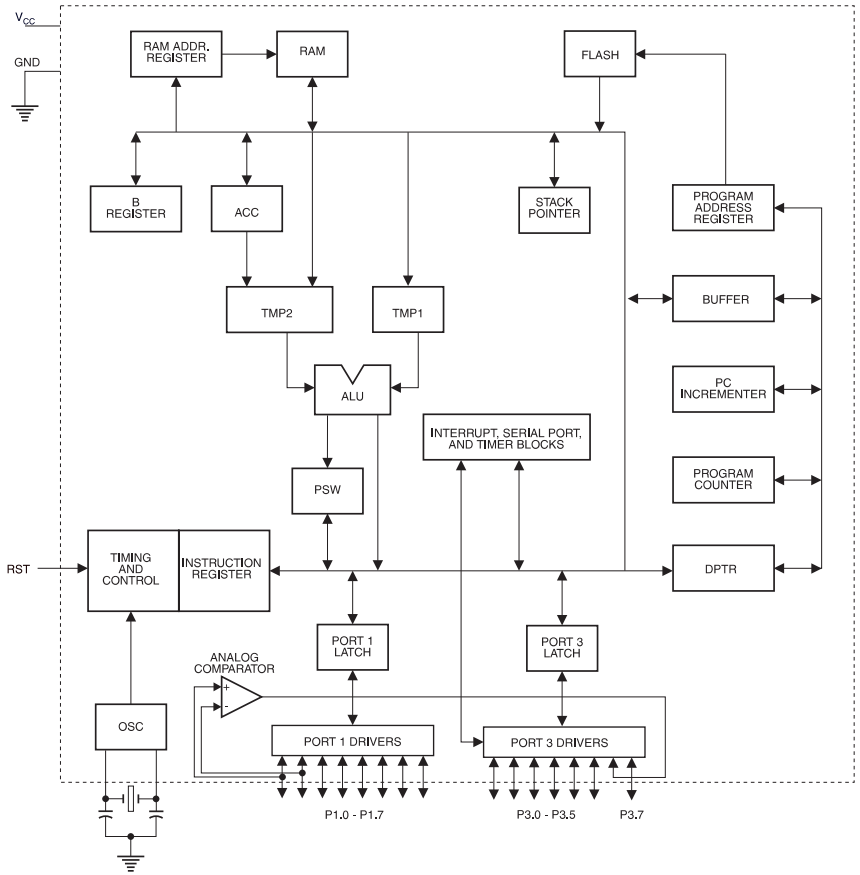


8-Bit Microcontroller with 4K Bytes Flash

AT89C4051 Preliminary

Rev. 1001A–02/98

Block Diagram



Pin Description

V_{CC}
Supply voltage.

GND
Ground.

Port 1
Port 1 is an 8-bit bidirectional I/O port. Port pins P1.2 to P1.7 provide internal pullups. P1.0 and P1.1 require external pullups. P1.0 and P1.1 also serve as the positive input (AIN0) and the negative input (AIN1), respectively, of the on-chip precision analog comparator. The Port 1 output buffers can sink 20 mA and can drive LED displays directly. When 1s are written to Port 1 pins, they can be used as inputs. When pins P1.2 to P1.7 are used as inputs and are externally pulled low, they will source current (I_{IL}) because of the internal pullups.

Port 1 also receives code data during Flash programming and verification.

Port 3
Port 3 pins P3.0 to P3.5, P3.7 are seven bidirectional I/O pins with internal pullups. P3.6 is hard-wired as an input to the output of the on-chip comparator and is not accessible as a general purpose I/O pin. The Port 3 output buffers can sink 20 mA. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C4051 as listed below:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)

Port 3 also receives some control signals for Flash programming and verification.

RST
Reset input. All I/O pins are reset to 1s as soon as RST goes high. Holding the RST pin high for two machine cycles while the oscillator is running resets the device.

Each machine cycle takes 12 oscillator or clock cycles.

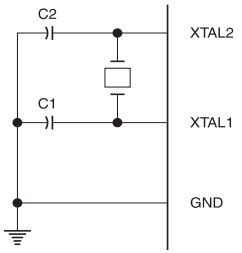
XTAL1
Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2
Output from the inverting oscillator amplifier.

Oscillator Characteristics

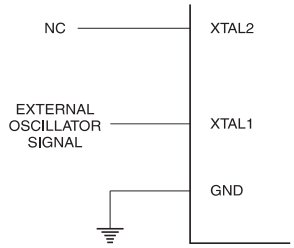
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 1. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in the table below.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

Table 1. AT89C4051 SFR Map and Reset Values

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H									0CFH
0C0H									0C7H
0B8H	IP XXX00000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0XX00000								0AFH
0A0H									0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H		SP 00000111	DPL 00000000	DPH 00000000				PCON 0XXX0000	87H

Restrictions on Certain Instructions

The AT89C4051 is an economical and cost-effective member of Atmel's growing family of microcontrollers. It contains 4K bytes of flash program memory. It is fully compatible with the MCS-51 architecture, and can be programmed using the MCS-51 instruction set. However, there are a few considerations one must keep in mind when utilizing certain instructions to program this device.

All the instructions related to jumping or branching should be restricted such that the destination address falls within the physical program memory space of the device, which is 4K for the AT89C4051. This should be the responsibility of the software programmer. For example, LJMP 0FE0H would be a valid instruction for the AT89C4051 (with 4K of memory), whereas LJMP 1000H would not.

1. Branching instructions:
LCALL, LJMP, ACALL, AJMP, SJMP, JMP @A+DPTR

These unconditional branching instructions will execute correctly as long as the programmer keeps in mind that the destination branching address must fall within the physical boundaries of the program memory size (locations 00H to FFFH for the 89C4051). Violating the physical space limits may cause unknown program behavior.

CJNE [...], DJNZ [...], JB, JNB, JC, JNC, JBC, JZ, JNZ With these conditional branching instructions the same rule above applies. Again, violating the memory boundaries may cause erratic execution.

For applications involving interrupts the normal interrupt service routine address locations of the 80C51 family architecture have been preserved.

2. MOVX-related instructions, Data Memory:
The AT89C4051 contains 128 bytes of internal data memory. Thus, in the AT89C4051 the stack depth is limited to 128 bytes, the amount of available RAM. External DATA memory access is not supported in this device, nor is external PROGRAM memory execution. Therefore, no MOVX [...] instructions should be included in the program.

A typical 80C51 assembler will still assemble instructions, even if they are written in violation of the restrictions mentioned above. It is the responsibility of the controller user to know the physical features and limitations of the device being used and adjust the instructions used correspondingly.

Program Memory Lock Bits

On the chip are two lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

Lock Bit Protection Modes(1)

Program Lock Bits			
	LB1	LB2	Protection Type
1	U	U	No program lock features.
2	P	U	Further programming of the Flash is disabled.
3	P	P	Same as mode 2, also verify is disabled.

Note: 1. The Lock Bits can only be erased with the Chip Erase operation.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

P1.0 and P1.1 should be set to '0' if no external pullups are used, or set to '1' if external pullups are used.

It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

Power Down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

P1.0 and P1.1 should be set to '0' if no external pullups are used, or set to '1' if external pullups are used.

PROGRAMMER'S GUIDE AND INSTRUCTION SET

Memory Organization

Program Memory

The 80C51 has separate address spaces for program and data memory. The Program memory can be up to 64k bytes long. The lower 4k can reside on-chip. Figure 1 shows a map of the 80C51 program memory.

The 80C51 can address up to 64k bytes of data memory to the chip. The MOVX instruction is used to access the external data memory.

The 80C51 has 128 bytes of on-chip RAM, plus a number of Special Function Registers (SFRs). The lower 128 bytes of RAM can be accessed either by direct addressing (MOV data addr) or by indirect addressing (MOV @Ri). Figure 2 shows the Data Memory organization.

Direct and Indirect Address Area

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into three segments as listed below and shown in Figure 3.

1. Register Banks 0-3: Locations 0 through 1FH (32 bytes). The device after reset defaults to register bank 0. To use the other register banks, the user must select them in software. Each register bank contains eight 1-byte registers 0 through 7. Reset initializes the stack pointer to location 07H, and it is incremented once to start from location 08H, which is the first register (R0) of the second register bank. Thus, in order to use more than one register bank, the SP should be initialized to a different location of the RAM where it is not used for data storage (i.e., the higher part of the RAM).
2. Bit Addressable Area: 16 bytes have been assigned for this segment, 20H-2FH. Each one of the 128 bits of this segment can be directly addressed (0-7FH). The bits can be referred to in two ways, both of which are acceptable by most assemblers. One way is to refer to their address (i.e., 0-7FH). The other way is with reference to bytes 20H to 2FH. Thus, bits 0-7 can also be referred to as bits 20.0-20.7, and bits 8-FH are the same as 21.0-21.7, and so on. Each of the 16 bytes in this segment can also be addressed as a byte.
3. Scratch Pad Area: 30H through 7FH are available to the user as data RAM. However, if the stack pointer has been initialized to this area, enough bytes should be left aside to prevent SP data destruction.

Figure 2 shows the different segments of the on-chip RAM.

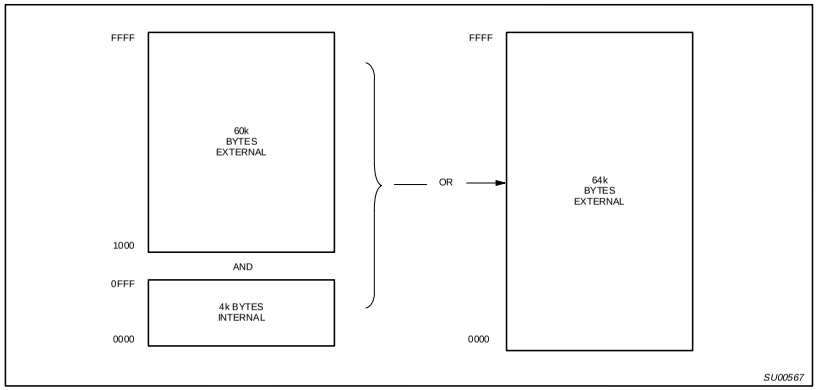


Figure 1. 80C51 Program Memory

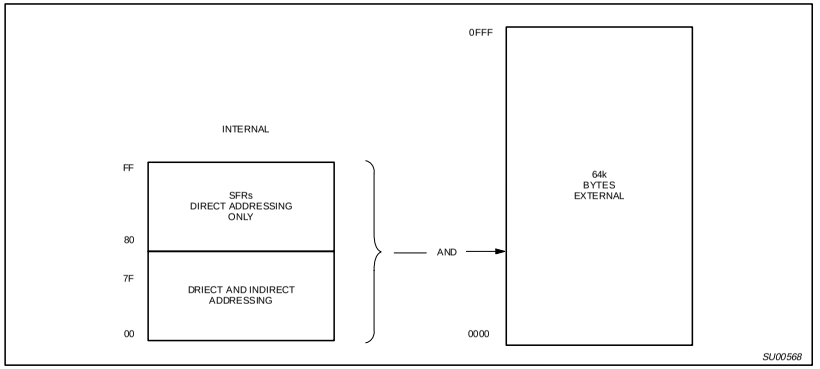


Figure 2. 80C51 Data Memory

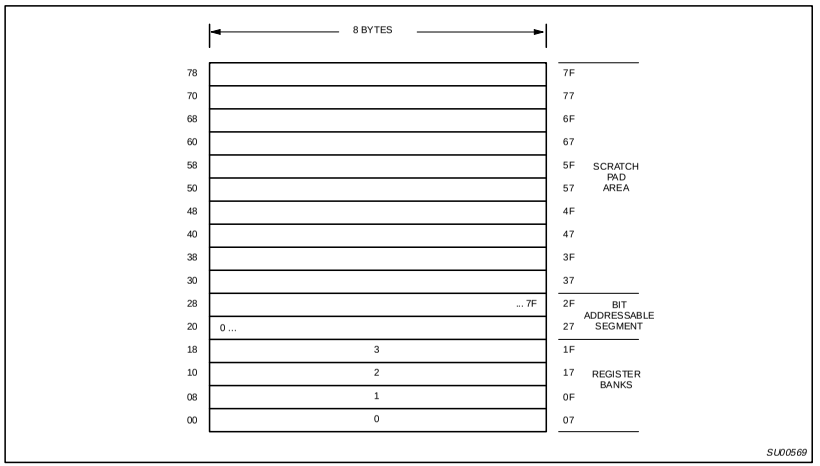


Figure 3. 128 Bytes of RAM Direct and Indirect Addressable

Table 1. 80C51 Special Function Registers

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB							LSB	RESET VALUE
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
DPTR	Data pointer (2 bytes)										
DPH	Data pointer high	83H									00H
DPL	Data pointer low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt enable	A8H	EA	–	–	ES	ET1	EX1	ET0	EX0	0x000000B
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt priority	B8H	–	–	–	PS	PT1	PX1	PT0	PX0	xx000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	–	–	–	–	–	–	T2EX	T2	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	A15	A14	A13	A12	A11	A10	A9	A8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1	T0	$\overline{INT1}$	$\overline{INT0}$	TxD	Rxd	FFH
PCON[1]	Power control	87H	SMOD	–	–	–	GF1	GF0	PD	IDL	0xxxxxxxB
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program status word	D0H	CY	AC	F0	RS1	RS0	OV	–	P	00H
SBUF	Serial data buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
SCON*	Serial controller	98H	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
TH0	Timer high 0	8CH									00H
TH1	Timer high 1	8DH									00H
TL0	Timer low 0	8AH									00H
TL1	Timer low 1	8BH									00H
TMOD	Timer mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H

NOTES:
* Bit addressable
1. Bits GF1, GF0, PD, and IDL of the PCON register are not implemented on the NMOS 8051/8031.

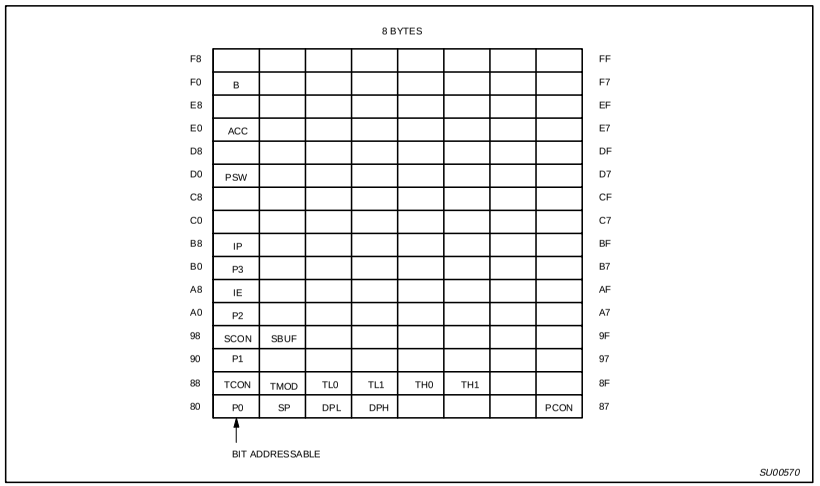


Figure 4. SFR Memory Map

Those SFRs that have their bits assigned for various functions are listed in this section. A brief description of each bit is provided for quick reference. For more detailed information refer to the Architecture Chapter of this book.

PSW: PROGRAM STATUS WORD. BIT ADDRESSABLE.

CY	AC	F0	RS1	RS0	OV	–	P

CY	PSW.7	Carry Flag.
AC	PSW.6	Auxiliary Carry Flag.
F0	PSW.5	Flag 0 available to the user for general purpose.
RS1	PSW.4	Register Bank selector bit 1 (SEE NOTE 1).
RS0	PSW.3	Register Bank selector bit 0 (SEE NOTE 1).
OV	PSW.2	Overflow Flag.
–	PSW.1	Usable as a general purpose flag.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of '1' bus in the accumulator.

NOTE:
1. The value presented by RS0 and RS1 selects the corresponding register bank.

RS1	RS0	REGISTER BANK	ADDRESS
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

PCON: POWER CONTROL REGISTER. NOT BIT ADDRESSABLE.

SMOD	–	–	–	GF1	GF0	PD	IDL

SMOD	Double baud rate bit. If Timer 1 is used to generate baud rate and SMOD = 1, the baud rate is doubled when the Serial Port is used in modes 1, 2, or 3.
–	Not implemented, reserved for future use.*
–	Not implemented reserved for future use.*
–	Not implemented reserved for future use.*
GF1	General purpose flag bit.
GF0	General purpose flag bit.
PD	Power Down Bit. Setting this bit activates Power Down operation in the 80C51. (Available only in CMOS.)
IDL	Idle mode bit. Setting this bit activates Idle Mode operation in the 80C51. (Available only in CMOS.)

If 1s are written to PD and IDL at the same time, PD takes precedence.

* User software should not write 1s to reserved bits. These bits may be used in future 8051 products to invoke new features.

Philips Semiconductors

80C51 Family

80C51 family programmer's guide and instruction set

INTERRUPTS:

To use any of the interrupts in the 80C51 Family, the following three steps must be taken.

1. **Set the EA (enable all) bit in the IE register to 1.**
2. **Set the corresponding individual interrupt enable bit in the IE register to 1.**
3. **Begin the interrupt service routine at the corresponding Vector Address of that interrupt. See Table below.**

INTERRUPT SOURCE	VECTOR ADDRESS
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI & TI	0023H

In addition, for external interrupts, pins INT0 and INT1 (P3.2 and P3.3) must be set to 1, and depending on whether the interrupt is to be level or transition activated, bits IT0 or IT1 in the TCON register may need to be set to 1.

ITx = 0 level activated

ITx = 1 transition activated

IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

EA	–	–	ES	ET1	EX1	ET0	EX0

EA	IE.7	Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	Not implemented, reserved for future use.*
—	IE.5	Not implemented, reserved for future use.*
ES	IE.4	Enable or disable the serial port interrupt.
ET1	IE.3	Enable or disable the Timer 1 overflow interrupt.
EX1	IE.2	Enable or disable External Interrupt 1.
ET0	IE.1	Enable or disable the Timer 0 overflow interrupt.
EX0	IE.0	Enable or disable External Interrupt 0.

* User software should not write 1s to reserved bits. These bits may be used in future 80C51 products to invoke new features.

ASSIGNING HIGHER PRIORITY TO ONE OR MORE INTERRUPTS:

In order to assign higher priority to an interrupt the corresponding bit in the IP register must be set to 1.

Remember that while an interrupt service is in progress, it cannot be interrupted by a lower or same level interrupt.

PRIORITY WITHIN LEVEL:

Priority within level is only to resolve simultaneous requests of the same priority level.

From high to low, interrupt sources are listed below:

IE0
TF0
IE1
TF1
RI or TI

IP: INTERRUPT PRIORITY REGISTER. BIT ADDRESSABLE.

If the bit is 0, the corresponding interrupt has a lower priority and if the bit is 1 the corresponding interrupt has a higher priority.

–	–	–	PS	PT1	PX1	PT0	PX0

–	IP.7	Not implemented, reserved for future use.*
–	IP.6	Not implemented, reserved for future use.*
–	IP.5	Not implemented, reserved for future use.*
PS	IP.4	Defines the Serial Port interrupt priority level.
PT1	IP.3	Defines the Timer 1 interrupt priority level.
PX1	IP.2	Defines External Interrupt 1 priority level.
PT0	IP.1	Defines the Timer 0 interrupt priority level.
PX0	IP.0	Defines the External Interrupt 0 priority level.

* User software should not write 1s to reserved bits. These bits may be used in future 80C51 products to invoke new features.

Philips Semiconductors

80C51 Family

80C51 family programmer's guide and instruction set

TCON: TIMER/COUNTER CONTROL REGISTER. BIT ADDRESSABLE.

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1	TCON.7	Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine.
TR1	TCON.6	Timer 1 run control bit. Set/cleared by software to turn Timer/Counter 1 ON/OFF.
TF0	TCON.5	Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine.
TR0	TCON.4	Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF.
IE1	TCON.3	External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed.
IT1	TCON.2	Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.
IE0	TCON.1	External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed.
IT0	TCON.0	Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt.

TMOD: TIMER/COUNTER MODE CONTROL REGISTER. NOT BIT ADDRESSABLE.

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
		Timer 1				**Timer 0**	

GATE	When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).
C/$\overline{T}$	Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).
M1	Mode selector bit. (NOTE 1)
M0	Mode selector bit. (NOTE 1)

NOTE 1:

M1	M0		Operating Mode
0	0	0	13-bit Timer (8048 compatible)
0	1	1	16-bit Timer/Counter
1	0	2	8-bit Auto-Reload Timer/Counter
1	1	3	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standart Timer 0 control bits. TH0 is an8-bit Timer and is controlled by Timer 1 control bits.
1	1	3	(Timer 1) Timer/Counter 1 stopped.

TIMER SET-UP

Tables 2 through 5 give some values for TMOD which can be used to set up Timer 0 in different modes.

It is assumed that only one timer is being used at a time. If it is desired to run Timers 0 and 1 simultaneously, in any mode, the value in TMOD for Timer 0 must be ORed with the value shown for Timer 1 (Tables 5 and 6).

For example, if it is desired to run Timer 0 in mode 1 GATE (extemal control), and Timer 1 in mode 2 COUNTER, then the value that must be loaded into TMOD is 69H (09H from Table 2 ORed with 60H from Table 5).

Moreover, it is assumed that the user, at this point, is not ready to turn the timers on and will do that at a different point in the program by setting bit TRx (in TCON) to 1.

TIMER/COUNTER 0

Table 2. As a Timer:

MODE	TIMER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	08H
1	16-bit Timer	01H	09H
2	8-bit Auto-Reload	02H	0AH
3	Two 8-bit Timers	03H	0BH

Table 3. As a Counter:

MODE	COUNTER 0 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	04H	0CH
1	16-bit Timer	05H	0DH
2	8-bit Auto-Reload	06H	0EH
3	One 8-bit Counter	07H	0FH

NOTES:
1. The timer is turned ON/OFF by setting/clearing bit TR0 in the software.
2. The Timer is turned ON/OFF by the 1-to-0 transition on $\overline{INT0}$ (P3.2) when TR0 = 1 (hardware control).

Philips Semiconductors

80C51 Family

80C51 family programmer's guide and instruction set

TIMER/COUNTER 1

Table 4. As a Timer:

MODE	TIMER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	00H	80H
1	16-bit Timer	10H	90H
2	8-bit Auto-Reload	20H	A0H
3	Does not run	30H	B0H

Table 5. As a Counter:

MODE	COUNTER 1 FUNCTION	TMOD	
		INTERNAL CONTROL (NOTE 1)	EXTERNAL CONTROL (NOTE 2)
0	13-bit Timer	40H	C0H
1	16-bit Timer	50H	D0H
2	8-bit Auto-Reload	60H	E0H
3	Not available	–	–

NOTES:
1. The timer is turned ON/OFF by setting/clearing bit TR1 in the software.
2. The Timer is turned ON/OFF by the 1-to-0 transition on $\overline{INT1}$ (P3.2) when TR1 = 1 (hardware control).

SCON: SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial Port mode specifier. (NOTE 1)
SM1	SCON.6	Serial Port mode specifier. (NOTE 1)
SM2	SCON.5	Enables the multiprocessor communication feature in modes 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 6.)
REN	SCON.4	Set/Cleared by software to Enable/Disable reception.
TB8	SCON.3	The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.
RB8	SCON.2	In modes 2 & 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.

NOTE 1:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	Shift Register	$F_{OSC}/12$
0	1	1	8-bit UART	Variable
1	0	2	9-bit UART	$F_{OSC}/64$ or $F_{OSC}/32$
1	1	3	9-bit UART	Variable

SERIAL PORT SET-UP:

Table 6.

MODE	SCON	SM2 VARIATION
0 1 2 3	10H 50H 90H D0H	Single Processor Environment (SM2 = 0)
0 1 2 3	NA 70H B0H F0H	Multiprocessor Environment (SM2 = 1)

GENERATING BAUD RATES

Serial Port in Mode 0:

Mode 0 has a fixed baud rate which is 1/12 of the oscillator frequency. To run the serial port in this mode none of the Timer/Counters need to be set up. Only the SCON register needs to be defined.

$$\text{Baud Rate} = \frac{\text{Osc Freq}}{12}$$

Serial Port in Mode 1:

Mode 1 has a variable baud rate. The baud rate is generated by Timer 1.

USING TIMER/COUNTER 1 TO GENERATE BAUD RATES:

For this purpose, Timer 1 is used in mode 2 (Auto-Reload). Refer to Timer Setup section of this chapter.

$$\text{Baud Rate} = \frac{K \quad \text{Osc Freq}}{32 \quad 12 \quad [256 \quad (TH1)]}$$

If SMOD = 0, then K = 1.
If SMOD = 1, then K = 2 (SMOD is in the PCON register).

Most of the time the user knows the baud rate and needs to know the reload value for TH1.

$$TH1 \quad 256 \quad \frac{K \quad \text{Osc Freq}}{384 \quad \text{baud rate}}$$

TH1 must be an integer value. Rounding off TH1 to the nearest integer may not produce the desired baud rate. In this case, the user may have to choose another crystal frequency.

Since the PCON register is not bit addressable, one way to set the bit is logical ORing the PCON register (i.e., ORL PCON,#80H). The address of PCON is 87H.

SERIAL PORT IN MODE 2:

The baud rate is fixed in this mode and is 1/32 or 1/64 of the oscillator frequency, depending on the value of the SMOD bit in the PCON register.

In this mode none of the Timers are used and the clock comes from the internal phase 2 clock.

SMOD = 1, Baud Rate = 1/32 Osc Freq.

SMOD = 0, Baud Rate = 1/64 Osc Freq.

To set the SMOD bit: ORL PCON,#80H. The address of PCON is 87H.

SERIAL PORT IN MODE 3:

The baud rate in mode 3 is variable and sets up exactly the same as in mode 1.

80C51 FAMILY INSTRUCTION SET

Table 7. 80C51 Instruction Set Summary

Interrupt Response Time: Refer to Hardware Description Chapter.

Instructions that Affect Flag Settings[1]

Instruction	Flag C	Flag OV	Flag AC	Instruction	Flag C	Flag OV	Flag AC
ADD	X	X	X	CLR C	0		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	0	X		ANL C,/bit	X		
DIV	0	X		ORL C,bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

[1]Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

Notes on instruction set and addressing modes:

Rn	Register R7-R0 of the currently selected Register Bank.
direct	8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a SFR [i.e., I/O port, control register, status register, etc. (128-255)].
@Ri	8-bit internal data RAM location (0-255) addressed indirectly through register R1 or R0.
#data	8-bit constant included in the instruction.
#data 16	16-bit constant included in the instruction
addr 16	16-bit destination address. Used by LCALL and LJMP. A branch can be anywhere within the 64k-byte Program Memory address space.
addr 11	11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2k-byte page of program memory as the first byte of the following instruction.
rel	Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is –128 to +127 bytes relative to first byte of the following instruction.
bit	Direct Addressed bit in Internal Data RAM or Special Function Register.

MNEMONIC		DESCRIPTION	BYTE	OSCILLATOR PERIOD
ARITHMETIC OPERATIONS				
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with carry	1	12
ADDC	A,#data	Add immediate data to A_{CC} with carry	2	12
SUBB	A,Rn	Subtract Register from A_{CC} with borrow	1	12
SUBB	A,direct	Subtract direct byte from A_{CC} with borrow	2	12
SUBB	A,@Ri	Subtract indirect RAM from A_{CC} with borrow	1	12
SUBB	A,#data	Subtract immediate data from A_{CC} with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12

Table 7. 80C51 Instruction Set Summary (Continued)

MNEMONIC		DESCRIPTION	BYTE	OSCILLATOR PERIOD
ARITHMETIC OPERATIONS (Continued)				
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment indirect RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A and B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12
LOGICAL OPERATIONS				
ANL	A,Rn	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12
RL	A	Rotate Accumulator left	1	12
RLC	A	Rotate Accumulator left through the carry	1	12
RR	A	Rotate Accumulator right	1	12
RRC	A	Rotate Accumulator right through the carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,Rn	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@Ri	Move indirect RAM to Accumulator	1	12

Table 7. 80C51 Instruction Set Summary (Continued)

MNEMONIC		DESCRIPTION	BYTE	OSCILLATOR PERIOD
DATA TRANSFER (Continued)				
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	Rn,A	Move Accumulator to register	1	12
MOV	Rn,direct	Move direct byte to register	2	24
MOV	RN,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,Rn	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@Ri	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@Ri,A	Move Accumulator to indirect RAM	1	12
MOV	@Ri,direct	Move direct byte to indirect RAM	2	24
MOV	@Ri,#data	Move immediate data to indirect RAM	2	12
MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to A_{CC}	1	24
MOVC	A,@A+PC	Move Code byte relative to PC to A_{CC}	1	24
MOVX	A,@Ri	Move external RAM (8-bit addr) to A_{CC}	1	24
MOVX	A,@DPTR	Move external RAM (16-bit addr) to A_{CC}	1	24
MOVX	A,@Ri,A	Move A_{CC} to external RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move A_{CC} to external RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,Rn	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@Ri	Exchange low-order digit indirect RAM with A_{CC}	1	12
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to carry	2	24
ANL	C,/bit	AND complement of direct bit to carry	2	24
ORL	C,bit	OR direct bit to carry	2	24
ORL	C,/bit	OR complement of direct bit to carry	2	24
MOV	C,bit	Move direct bit to carry	2	12
MOV	bit,C	Move carry to direct bit	2	24
JC	rel	Jump if carry is set	2	24
JNC	rel	Jump if carry not set	2	24

Philips Semiconductors

80C51 Family

80C51 family programmer's guide and instruction set

Table 7. 80C51 Instruction Set Summary (Continued)

MNEMONIC		DESCRIPTION	BYTE	OSCILLATOR PERIOD
BOOLEAN VARIABLE MANIPULATION (Continued)				
JB	rel	Jump if direct bit is set	3	24
JNB	rel	Jump if direct bit is not set	3	24
JBC	bit,rel	Jump if direct bit is set and clear bit	3	24
PROGRAM BRANCHING				
ACALL	addr11	Absolute subroutine call	2	24
LCALL	addr16	Long subroutine call	3	24
RET		Return from subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute jump	2	24
LJMP	addr16	Long jump	3	24
SJMP	rel	Short jump (relative addr)	2	24
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is zero	2	24
JNZ	rel	Jump if Accumulator is not zero	2	24
CJNE	A,direct,rel	Compare direct byte to A_{CC} and jump if not equal	3	24
CJNE	A,#data,rel	Compare immediate to A_{CC} and jump if not equal	3	24
CJNE	RN,#data,rel	Compare immediate to register and jump if not equal	3	24
CJNE	@Ri,#data,rel	Compare immediate to indirect and jump if not equal	3	24
DJNZ	Rn,rel	Decrement register and jump if not zero	2	24
DJNZ	direct,rel	Decrement direct byte and jump if not zero	3	24
NOP		No operation	1	12

APPENDIX C: ALPHABETICAL INDEX

SYMBOLS

A

B

C

D

S

T

W